**INVESTIGATION OF COOPERATIVE BEHAVIOR IN AUTONOMOUS WIDE AREA SEARCH MUNITIONS**

THESIS

Robert E. Dunkel III, Captain, USAF

AFIT/GAE/ENY/02-4

**DEPARTMENT OF THE AIR FORCE**
**AIR UNIVERSITY**

# AIR FORCE INSTITUTE OF TECHNOLOGY

**Wright-Patterson Air Force Base, Ohio**

# Report Documentation Page

| Report Date | Report Type | Dates Covered (from... to) |
|---|---|---|
| 26 Mar 2002 | Final | Sep 2000 - Mar 2002 |

| | |
|---|---|
| **Title and Subtitle**<br>Investigation of Cooperative Behavior in Autonomous Wide Search Munitions | **Contract Number** |
| | **Grant Number** |
| | **Program Element Number** |
| **Author(s)**<br>Capt Robert E. Dunkel, III, USAF | **Project Number** |
| | **Task Number** |
| | **Work Unit Number** |
| **Performing Organization Name(s) and Address(es)**<br>Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 P Street, Bldg 640 WPAFB, OH 45433-7765 | **Performing Organization Report Number**<br>AFIT/GA/ENY/02-4 |
| **Sponsoring/Monitoring Agency Name(s) and Address(es)**<br>Mr. Rob Murphey AFRL/MNGN 101 W. Eglin Blvd., Ste 330 Eglin AFB, FL 32542-6810 | **Sponsor/Monitor's Acronym(s)** |
| | **Sponsor/Monitor's Report Number(s)** |
| **Distribution/Availability Statement**<br>Approved for public release, distribution unlimited | |
| **Supplementary Notes**<br>The original document contains color images. | |

**Abstract**
The purpose of this research is to investigate the effectiveness of wide-area search munitions in various scenarios using different cooperative behavior algorithms. The general scenario involves multiple autonomous munitions searching for an unknown number of targets of different priority in unknown locations. Three cooperative behavior algorithms are used in each scenario: no cooperation, cooperative attack only, and cooperative classification and attack. In the cooperative cases, the munitions allocate tasks on-line as a group, using linear programming techniques to determine the optimum allocation. Each munition provides inputs to the task allocation routine in the form of probabilities of successfully being able to complete the various tasks. These probabilities of success are based on statistical Poisson field theory. Weighting parameters are applied to the probabilities of success so that optimum settings can be determined via Response Surface Methodology. Results are compared within and across the various scenarios. Initial results did not reflect expected behavior (due to poor choice of responses to optimize). Experiments were modified and more desirable results obtained. In general, cooperative engagement alone attacks and kills fewer targets than no cooperation. Cooperative classification however, kills fewer targets at low false target attack rates (< 0.005/sq km), but outperforms the other algorithms as the false target attack rate increases. This is due primarily to the fact that cooperative classification significantly reduces and stabilizes the effective false target attack rate.

**Subject Terms**
Cooperative Engagement, Cooperative Behavior, Autonomous Munitions, Wide Area Search Munitions

| **Report Classification** | **Classification of this page** |
|---|---|
| unclassified | unclassified |
| **Classification of Abstract** | **Limitation of Abstract** |
| unclassified | UU |

**Number of Pages**
103

AFIT/GAE/ENY/02-4

INVESTIGATION OF COOPERATIVE BEHAVIOR IN AUTONOMOUS WIDE AREA SEARCH MUNITIONS

THESIS

Presented to the Faculty

Department of Aeronautical and Astronautical Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aeronautical Engineering

Robert E. Dunkel III, BS

Captain, USAF

March 2002

AFIT/GAE/ENY/02-4

INVESTIGATION OF COOPERATIVE BEHAVIOR IN AUTONOMOUS WIDE AREA SEARCH MUNITIONS

Robert E. Dunkel III, BS

Captain, USAF

Approved:

| | |
|---|---|
| Lt Col David Jacques, Ph.D. (Chairman) | 12 MAR 02<br>Date |
| Lt Col Price Smith, Ph.D. (Member) | 12 MAR 02<br>Date |
| Lt Col Ray Hill, Ph.D. (Member) | 12 Mar 02<br>Date |

# Acknowledgements

(This page intentionally left blank)

# Table of Contents

Page

Page

# List of Figures

# List of Tables

# List of Abbreviations and Symbols

| | |
|---|---|
| $\alpha$ | False Target Attack Rate |
| $\beta$ | Weighting parameter for the value of low priority targets relative to high priority targets |
| $\xi$ | Weighting parameter for continued search versus attack |
| $\eta_t$ | Target Density |
| $\eta_{total\_tgts}$ | Sum of the individual target type densities |
| $\eta_{FT}$ | False Target Density |
| A | Area already searched by a munition |
| $A_s$ | Search Area |
| AFB | Air Force Base |
| AFIT | Air Force Insitute of Technology |
| AFRL | Air Force Research Laboratory |
| ATR | Autonomous Target Recognition |
| BDA | Battle Damage Assessment |
| CCD | Central Composite Design |
| dA | Differential increase in the area searched by a munition (A) |
| ETA | Estimated Time of Arrival—time required for a munition to reach a target |
| $N_{tgts}$ | Number of targets |
| $P_E$ | Probability of Encountering a true target |
| $P_{\overline{FA}}$ | Probability of not False Alarming |
| $P_{FTA\|E}$ | Probability of a false target attack given a false target encounter |

| | |
|---|---|
| $P_k$ | Probability of Kill |
| $P_{\overline{RT}}$ | Probability of not recognizing a target |
| $P_{RT\|TR}$ | Probability of a real target given that a target report is made |
| $P_{sa}$ | Probability of a successful attack |
| $P_{ss}$ | Probability of a successful search |
| $P_{TR}$ | Probability of target report |
| RSM | Response Surface Methodology |
| $t_{ETA}$ | Time required for a munition to attempt an attack, abort, and return to searching |
| $t_r$ | Time remaining |
| V | Munition velocity |
| VACA | Vehicles directorate of the Air Force Research Laboratory (AFRL/VACA) |
| W | Width of a munition's sensor footprint |

# Abstract

The purpose of this research is to investigate the effectiveness of wide-area search munitions in various scenarios using different cooperative behavior algorithms. The general scenario involves multiple autonomous munitions searching for an unknown number of targets of different priority in unknown locations. Three cooperative behavior algorithms are used in each scenario: no cooperation, cooperative attack only, and cooperative classification and attack. In the cooperative cases, the munitions allocate tasks on-line as a group, using linear programming techniques to determine the optimum allocation. Each munition provides inputs to the task allocation routine in the form of probabilities of successfully being able to complete the various tasks. These probabilities of success are based on statistical Poisson field theory. Weighting parameters are applied to the probabilities of success so that optimum settings can be determined via Response Surface Methodology.

Results are compared within and across the various scenarios. Initial results did not reflect expected behavior (due to poor choice of responses to optimize). Experiments were modified and more desirable results obtained. In general, cooperative engagement alone attacks and kills fewer targets than no cooperation. Cooperative classification however, kills fewer targets at low false target attack rates (< 0.005/km$^2$), but outperforms the other algorithms as the false target attack rate increases. This is due primarily to the fact that cooperative classification significantly reduces and stabilizes the effective false target attack rate.

INVESTIGATION OF COOPERATIVE BEHAVIOR IN AUTONOMOUS WIDE AREA SEARCH MUNITIONS

# I. Introduction

## *1.1 General*

The purpose of this research is to investigate the effectiveness of wide-area search munitions in various scenarios using different cooperative behavior algorithms. The general scenario involves multiple autonomous munitions searching for an unknown number of targets of different priority in unknown locations. When one munition finds and identifies a target, the information is communicated to the other munitions which otherwise would not have any knowledge of the target location or identification. Together (i.e. cooperatively) the munitions make a decision as to whether the target should be attacked and if so, which munition should attack it. An attack on a target does not guarantee a kill however, so each munition keeps a log of identified targets and continuously has to decide whether to attack a known target or to continue searching. How the munitions make this decision is the thrust of this research.

The Munitions Directorate of the Air Force Research Laboratory at Eglin Air Force Base (AFB) sponsored this research. All research was conducted at the Air Force Institute of Technology (AFIT), Wright-Patterson AFB, Ohio.

## 1.2 Background

In order to remain the world's premier air power, the United States Air Force continually searches for ways to improve its warfighting capabilities. Recent attention has focused on improving not only mission success but also mission efficiency, the question being "How many targets can be killed in a sortie?" rather than "How many sorties are required to kill a target?". The paradigm shift, while perhaps subtle, is significant. It implies that each sortie employs multiple weapons, and that the weapons themselves are highly effective in killing targets. Thus, emphasis has been placed on small weapons (so that many can be carried in a single sortie) with high lethality.

High lethality can be achieved in a number of ways. Typically, high lethality is realized with large warheads, but this competes with the objective of making small weapons. An alternative is to create small weapons with small warheads and highly sophisticated guidance packages, thereby allowing the weapon to precisely hit a vulnerable point on a target. This however, requires a level of sophistication that may be beyond current technology or cost effectiveness. Yet another alternative (or perhaps an augmentation to weapons with sophisticated guidance packages) may be to use small weapons that behave cooperatively. Small, low cost, lightweight, autonomous weapons that can recognize targets are already in development by the Air Force and its defense contractors. Individually, these weapons are not as lethal as larger, more expensive weapons, but if the weapons can make use of cooperative behavior, perhaps their decreased individual capabilities can be overcome.

The term cooperative behavior is itself a broad description of a host of subject areas including, but not limited to, communication, self-organization, and task allocation.

Each of these areas is discussed in more detail below. The study of cooperative behavior spans several disciplines, but research in the areas of ethology (the study of animal behavior) and robotics appears to provide material that is the most pertinent to the current research. Many of the examples in the following discussion will be from these disciplines.

**1.2.1 Communication.** In all examples of cooperative behavior, there is some form of communication. The nature of the communication however, is widely varied. In his study considering the behavior of flocks, herds, and schools, Reynolds postulates that each agent responds only to the actions of nearby agents (11). Thus communications are local, but they are not actively broadcast. Individual agents react to information they perceive and those reactions force other agents to react. A slightly more complex system is employed by ant colonies (1). When an individual ant locates a food source, it returns to the colony, laying down a pheromone trail of decreasing intensity in the process. Other ants in the colony simply follow the increasing pheromone gradient to the food source and at the same time deposit their own pheromone trail. As more ants travel between colony and food source, the pheromone trail is reinforced. In this case, communications are still local (only ants close to the pheromone trail will know about it), but involve active transmission (depositing pheromones), resulting in an autocatalytic process. Stone and Veloso take an interesting approach to communication between cooperative robots (13). Their team of robots experiences periods of both local and global communications. In general, the individual agents are able to actively communicate with other agents near them. However, the agents periodically communicate on a more global scale to re-synchronize their efforts.

#### 1.2.2 Self-Organization.

A key part of cooperative behavior involves how individual agents organize themselves within the group. As already alluded to, Reynolds' research employs agents who simply react to changes in their surroundings. Kube's box-pushing robots use similar logic (8). Individual robots are programmed to avoid obstacles (such as other robots) while attempting to accomplish a common task (moving a box). This is perhaps the most primitive method of self-organization within a group, since each agent acts independently without considering the impact on the group as a whole. Other methods involve more complex planning schemes. In Stone and Veloso's research, individual agents can re-organize themselves into different formations and assign tasks to the various agents during the periods of global communication. Such logic provides a more flexible organizational structure, allowing the group to adjust to changes in the environment or goal.

Organizational decisions can also be made by considering factors such as possible (or expected) future task assignments or individual agent capabilities (in the case of dissimilar agents). While some of these considerations tie in more appropriately with the subject of task allocation, it is important to mention them here. The organization and re-organization of autonomous search weapons may have significant impact on mission success. For example, if an individual search weapon disappears (either from some sort of failure, being attacked by an enemy, or flying itself into a target), how should the other agents organize themselves? Should the remaining agents close the gap created by the missing agent, or should they simply continue to fly their present routes? The answer to this question is outside the scope of this research, but it is certainly pertinent to the discussion of cooperative behavior.

#### 1.2.3 Decision Making and Task Allocation.

Decision-making and task allocation are the primary areas of concern for this research. The discussion up until now presented various forms of communication and self-organization within a group, but what is the purpose of these actions, and more importantly, how do individual agents know what tasks to perform in order to achieve that purpose? Using simple common sense and observation, we can say that individual agents within a group appear to behave according to some set of rules. For ants, the rules seem simple: follow the steepest pheromone gradient. In his research on flocks, herds, and schools, Reynolds surmised that individual agents behave according to two basic principles: the desire to stay close to the flock and the desire to avoid collisions with other agents (11). Thus, he constructed an analytic model using attractive and repulsive forces to govern individual agent behavior.

Other behavior governing rules have been investigated for applications in autonomous munitions. Both Frelinger et al. (2) and Gillen (3) developed simulations wherein individual munitions cooperatively search for and engage targets. Individual agents determine their appropriate tasks based on decision rules which consider parameters such as distance to target, fuel remaining, range rate, etc. Typically a munition will search an area until the value of a particular decision rule exceeds a predetermined threshold. At that point, the munition leaves the search pattern to accomplish the task associated with the particular decision rule. Overall, the research by Frelinger et al. and Gillen was quite good and yielded significant insight into cooperative behavior (in fact, this research builds substantially off of Gillen's research). However, there are weaknesses. One such weakness lies in the development of the decision rules. In both cases, the forms of the decision rules were arrived at heuristically, without

extensive theoretical development. While the decision rules worked well in their respective simulations, a strong theoretical basis may have provided more insight into significant parameters to include in the decision rules and under what scenarios the decision rules would be applicable.

In the above examples, individual agents do not consciously behave cooperatively. Each individual agent acts independently, according to some set of behavior rules. When an agent decides to perform a task, it is because it is most advantageous (or rule conforming) for that individual agent to perform the task. The fact that the task allocation works well for the group is simply a byproduct of the behavior rule development. If properly developed, the rules governing individual behavior will also maximize the entire group's effectiveness. For animals, one can assume that the behavior rules are either learned through experience or are ingrained at some instinctual level at birth. In Gillen's research, weighting parameters within the decision rule were optimized off-line based on the entire group's performance. Whatever the method behind the behavior rule development, the fact remains that the individual agents do not consciously allocate tasks based on what is best for the group. In some ways, this is an attractive feature of cooperative behavior. Without conscious thought or extensive communication, individual agents perform tasks that benefit the group at large. Such schemes work well for tasks such as foraging or finding the shortest path, where it is acceptable or even desirable for individual agents to perform the same tasks. However, there may be situations where such duplication of effort is wasteful. In these situations, it may be more beneficial for the agents to allocate tasks as a group—still following a set of rules, but a set of rules that govern group behavior rather than individual agent behavior.

Schumacher et al. implemented just such a rule framework (12). In their simulation (which is modified and used for this research), they allocated tasks using linear programming and optimization techniques. With this method, which is discussed more in Chapter III, each agent computes the benefits (to the individual agent) of performing certain tasks. Those benefits, in turn, are used to find an optimal solution to the capacitated transshipment problem (an optimization technique). The final task allocations are those that provide the greatest benefit to the system as a whole, rather than to individual agents. In some ways, this is an improvement over the cooperative behaviors previously discussed since tasks are allocated to maximize the performance of the entire group. A possible disadvantage to the method is increased communication and computational requirements between individual agents.

**1.2.4 Sensitivities in Cooperative Weapons.** Previous research by Gillen provides insight into sensitivities and limitations of cooperative munitions (3). Some of these limitations are addressed by this research and are introduced in this section.

Gillen's research shows that one important factor in the mission effectiveness of wide-area search munitions is the false target attack rate, $\alpha$. As $\alpha$ increases, overall mission effectiveness dramatically decreases. A conceptual method of reducing $\alpha$ in cooperative weapons is to have multiple munitions attempt to classify an object as a target or non-target. However, this may also have the undesirable effect of decreasing the identification rate of valid targets.

Some of the parameters within Gillen's decision rule carried much more weight than other parameters. Specifically, time of flight seemed to dominate the decision rule. This begs the question as to the source of the sensitivity. Is time of flight dominant

because it is *the* most important parameter for cooperative munitions, or is it dominant because the other parameters chosen for the decision rule were simply insignificant? The answer to this question is not easily arrived at since the decision rule does not have the theoretical basis supporting its final form.

Another limitation pointed out by Gillen, and perhaps related to the form of the decision rule, is a lack of robustness across various scenarios. Gillen optimized his decision rule for several different scenarios. He then investigated the robustness of the results by using optimized decision rules in scenarios other than that for which they were optimized. The outcome showed a general lack of robustness in results. This is a concern since actual battlefield parameters such as the number of targets, warhead lethality, etc. are not known with certainty. Gillen suggests investigating different forms of the decision rule to perhaps achieve better results.

## 1.3 Objectives

The purpose of this study is to investigate the effectiveness of wide-area search munitions in various scenarios using various behavior rules. More specific objectives are:

1. Develop a simulation that incorporates advantages as well as possible disadvantages of cooperative behavior.
2. Determine under what circumstances (munition and battlefield characteristics) it is beneficial to use cooperative behavior and under what circumstances it is detrimental to use cooperative behavior.
3. Determine the degree of benefit (if any) gained from cooperative behavior over non-cooperative behavior.

## 1.4 Approach and Scope

A computer simulation is used to model multiple unmanned wide area search munitions searching for and attacking randomly placed targets on a virtual battlefield. Three forms of cooperation are investigated for each scenario of battlefield characteristics. The first case is a non-cooperative case in which the munitions do not communicate with each other at all—each munition can attack only those targets it has independently identified. In the second case the munitions can cooperatively attack—each munition can attack any target identified by any munition. In the third case, the munitions cooperatively classify and attack targets—multiple passes (by one munition or multiple munitions) must be made over a target to better identify the target before allowing any munition to attack it. Multiplicative factors are applied to the decision rules (called benefit calculations in this simulation), and the values of the factors are optimized for each case in every scenario using Response Surface Methodology (RSM). The results from each case are compared within and across the various scenarios to determine general characteristics of cooperative behavior. The benefit calculations themselves are developed from statistical theory (see Chapter II) and have fewer weighting parameters (only one for each rule) than Gillen used in his research.

Eight munitions are employed in each scenario, and a variety of scenarios are explored. The scenarios themselves are defined by the number of targets and false targets, the distribution of targets, the lethality of the munitions, and other parameters. A complete test matrix listing all of the parameter combinations tested is included in Appendix A.

Inter-munition communications within this research are global and reliable (though not always accurate). Although this is obviously not truly representative of the real world, other issues took precedence to the communications issue, as discussed in section 3.1.6. Limited and unreliable communications are simply left as a recommendation for further research.

All targets and non-targets in the simulation are stationary. Moving targets complicate the simulation more than was desired for this research. This issue is also left as a recommendation for further research.

## *1.5 Relevance*

This research is conducted with a generic computer simulation that does not model any specific wide area search munition. Consequently, all conclusions drawn may be applied to any scenario with similar vehicle and battlefield characteristics. While the research is conducted under the context of weapons effectiveness, it is not limited to that application. Much of the theory presented in Chapter II is developed independent of the current application and has relevance to cooperative search in general. Likewise, the issues encountered in developing a realistic simulation and presented in Chapter III are pertinent to the development of any simulation of cooperative agents.

# II. Wide Area Search Munitions

## *2.1 The Single Munition, Single Target Case*

**2.1.1 General.** The most basic scenario involving autonomous munitions is the single munition, single target case. Suppose we want to evaluate the probability of successfully searching an area $A_s$. Let the width of the area equal the width of the munition's sensor footprint and the length of the area be much greater than the height of the sensor footprint. Within this area, there is a single true target along with a Poisson field of non-targets that the munition may misidentify as valid targets. As the munition searches area $A_s$, an already searched area, A, becomes apparent, as shown in Figure 2.1. For any discrete time interval, the area A will increase by dA, which is equal to the product of the time interval, vehicle velocity, and sensor footprint width.

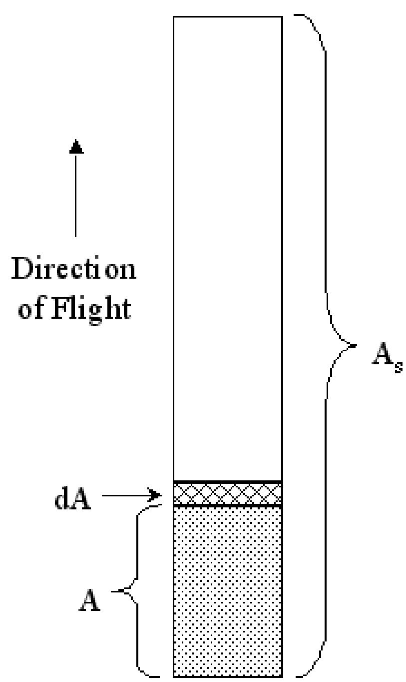



**Figure 2.1 Search Setup**

**2.1.2 Probability of Target Encounter.** As the munition searches area $A_s$, the incremental probability that it will encounter the target in dA can be obtained from Poisson probability distributions (6). In the case of a single true target and multiple false targets, the probability that the munition will encounter the target in area dA is the product of the probability that there are no false alarms ($P_{\overline{FA}}$) in the already searched area A and the probability that the target is encountered ($P_E$) in area dA. The analytic forms of these two probabilities are shown in equations 2.1 and 2.2:

$$P_{\overline{FA}} = e^{-\alpha A} \tag{2.1}$$

$$P_E = \eta_t \cdot dA \tag{2.2}$$

where $\eta_t$ is the target density and $\alpha$ is the false target attack rate. The false target attack rate is defined as the product of the average density of false targets ($\eta_{FT}$) and the probability that the munition will attack a false target that it has encountered ($P_{FTA|E}$):

$$\alpha = \eta_{FT} \cdot P_{FTA|E} \tag{2.3}$$

As previously mentioned, the incremental probability that the munition will encounter the target in dA is the product of equations 2.1 and 2.2:

$$\Delta P_E(A) = e^{-\alpha A} \cdot \eta_t \cdot dA \tag{2.4}$$

Equation 2.4 can be integrated over the entire area $A_s$ to yield:

$$P_E(A_s) = \eta_t \cdot (\frac{1 - e^{-\alpha A_s}}{\alpha}) \tag{2.5}$$

Since for the single target case the target density is simply the inverse of the area $A_s$, equation 2.5 simplifies to:

$$P_E(A_s) = \frac{1 - e^{-\alpha A_s}}{\alpha \cdot A_s} \tag{2.6}$$

If we define $A_s$ as the product of the vehicle velocity, the width of the search area, and the remaining search time ($t_r$), equation 2.6 can be written as:

$$P_E(t_r) = \frac{1 - e^{-\alpha \cdot V \cdot t_r \cdot W}}{\alpha \cdot V \cdot t_r \cdot W} \tag{2.7}$$

The importance of equation 2.7 will become apparent in section 2.1.4.

### 2.1.3 Probability of Target Report.

When the munition encounters the target, it may or may not recognize it as a target. This *a priori* probability of target report ($P_{TR}$) is a characteristic of the munition's autonomous target recognition (ATR) capability and is one measure of the performance of the ATR algorithm. For the single target case (or more appropriately, the single target type case), it can be expressed in terms of a matrix as shown in Table 2.1 (3)

**Table 2.1 Binary Confusion Matrix**

| | Encountered Object | |
|---|---|---|
| | Target | Non-Target |
| Declared Object | | |
| Target | $P_{TR}$ | $P_{FTA\|E}$ |
| Non-Target | $1\text{-}P_{TR}$ | $1\text{-}P_{FTA\|E}$ |

Table 2.1 is typically called a confusion matrix since it includes not only the probabilities of correctly identifying objects (the diagonals), but also the probabilities of misidentifying objects (the off-diagonals). Notice that there are only two parameters that define the matrix: $P_{FTA|E}$ and $P_{TR}$. For the single target case, these probabilities are drawn directly from the confusion matrix for use in calculations. In later sections we will see that these probabilities are not so straightforward.

**2.1.4 Outcome Trees.** The possible outcomes resulting from the munition's search of area $A_s$ and attack on any particular declared target in that area can be laid out in outcome trees and the likelihood of particular outcomes (both positive and negative) evaluated. For the single munition, single target case, these likelihoods are somewhat moot. Once the munition identifies a target, the best action for the munition to perform is attack—continued searching cannot yield any better results. However, the outcome trees still provide insight into the behavior of the system and are mentioned here for pedagogical purposes.

First, consider the possible outcomes of searching area $A_s$. The outcome tree for a search is shown in Figure 2.2 (4).

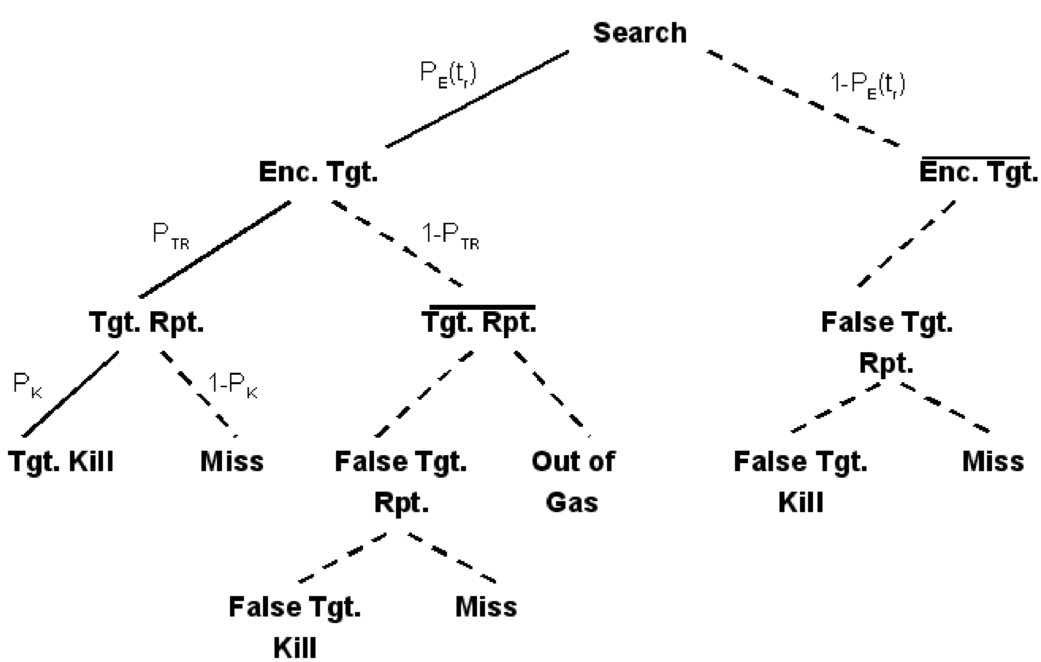



**Figure 2.2 Search Outcome Tree for Single Munition, Single Target Scenario**

In Figure 2.2, the solid lines represent positive outcomes and the dashed lines indicate negative outcomes. According to the outcome tree, a searching vehicle can either encounter the true target or not encounter the true target, the probabilities of which can be

calculated from equation 2.7. If the munition does not encounter the true target, then it must have encountered a false target, declared it as a real target, and attacked it—an undesirable outcome. On the other hand, if the munition encounters the true target, it may or may not report the target, according to the value of $P_{TR}$ in the confusion matrix. If it recognizes the target, the munition can execute an attack that may or may not be successful depending on the lethality of the warhead. If the munition does not recognize the target, then it continues to search with no possibility of ever finding the target (there is only one target and the search of $A_s$ is exhaustive but not duplicative). The munition will either false alarm or simply run out of gas. The likelihood of any particular outcome is simply the product of the probabilities along the path to that outcome. For the single target case, the probability of a successful search ($P_{ss}$) is represented solely by the left-most branch. The analytical formulation is (5):

$$P_{SS} = P_K \cdot P_{TR} \cdot P_E = P_K \cdot P_{TR} \cdot \frac{1 - e^{-\alpha \cdot V \cdot t_r \cdot W}}{\alpha \cdot V \cdot t_r \cdot W} \tag{2.8}$$

where $P_k$ is the probability that an attack on a target will result in a kill.

A similar outcome tree can be constructed for engaging a declared target, as shown in Figure 2.3:

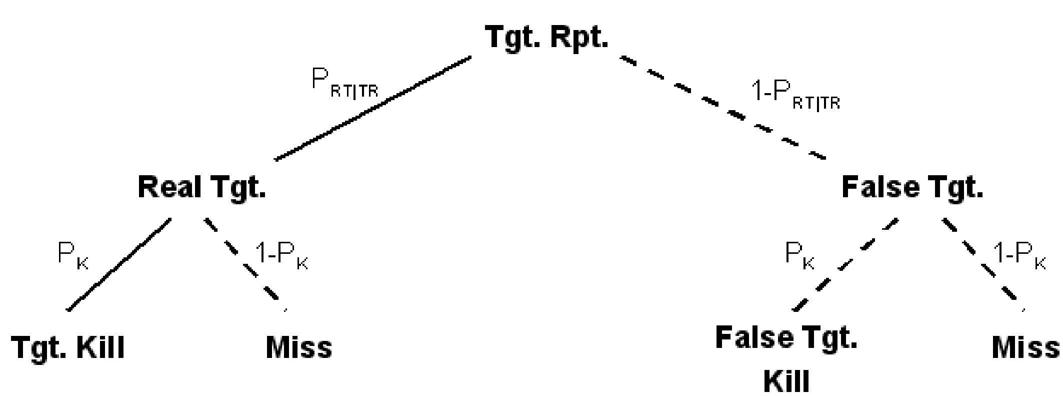



**Figure 2.3 Engagement Outcome Tree for Single Munition, Single Target Scenario**

This engagement tree is relatively simple—later cases become more complex. When the munition attacks a declared target, there is a probability less than one that the declared target is indeed the real target ($P_{RT|TR}$). This probability can be estimated by a simple ratio of the true target attack rate to the total attack rate (for the true target and false targets) as shown in equation 2.9 (5):

$$P_{RT|TR} = \frac{P_{TR} \cdot \eta_T}{P_{TR} \cdot \eta_T + P_{FTA|E} \cdot \eta_{FT}} \tag{2.9}$$

The probability of a successful attack is calculated in the same way as the probability of a successful search:

$$P_{sa} = P_{RT|TR} \cdot P_K \tag{2.10}$$

## *2.2 The Single Munition, Multi-Target Case*

**2.2.1 Probability of Target Encounter.** The single munition, multi-target case is set up similar to the single target case, with minor modifications. Refer to Figure 2.1. For the single target case, $\Delta P_E(A)$ was only dependent on the probability of not false alarming in A and the probability of encountering the target in dA (see equation 2.4). With a Poisson field of targets, $\Delta P_E(A)$ is also dependent on the probability of not previously recognizing a target ($P_{\overline{RT}}$) within A. This can be obtained from a Poisson probability distribution:

$$P_{\overline{RT}} = e^{-\eta_t P_{TR} A} \tag{2.11}$$

Thus, the incremental probability of encountering a target in dA becomes the product of equations 2.1, 2.2, and 2.11:

$$\Delta P_E(A) = e^{-(\eta_t P_{TR} + \alpha) A} \cdot \eta_t \cdot dA \tag{2.12}$$

This can be integrated over the entire search area to obtain:

$$P_E(A_S) = \frac{\eta_t}{\eta_t P_{TR} + \alpha} \cdot (1 - e^{-(\eta_t P_{TR} + \alpha) A_s}) \tag{2.13}$$

**2.2.2 Probability of Target Report.** As with the single munition, single target case, $P_{TR}$ is drawn from the confusion matrix. However, the complexity of the confusion matrix may vary greatly depending on the variety of target types available to the ATR algorithm. Suppose an ATR is capable of distinguishing four different target types. The confusion matrix would look similar to Table 2.2 (3):

**Table 2.2 Multiple Target Type Confusion Matrix**

| Encountered Object | | | | | |
|---|---|---|---|---|---|
| | Tgt Type 1 | Tgt Type 2 | Tgt Type 3 | Tgt Type 4 | Non-Target |
| Declared Object | | | | | |
| Tgt Type 1 | $P_{TR\ 1\|Type\ 1}$ | $P_{TR\ 1\|Type\ 2}$ | $P_{TR\ 1\|Type\ 3}$ | $P_{TR\ 1\|Type\ 4}$ | $P_{FTA1\|E}$ |
| Tgt Type 2 | $P_{TR\ 2\|Type\ 1}$ | $P_{TR\ 2\|Type\ 2}$ | $P_{TR\ 2\|Type\ 3}$ | $P_{TR\ 2\|Type\ 4}$ | $P_{FTA2\|E}$ |
| Tgt Type 3 | $P_{TR\ 3\|Type\ 1}$ | $P_{TR\ 3\|Type\ 2}$ | $P_{TR\ 3\|Type\ 3}$ | $P_{TR\ 3\|Type\ 4}$ | $P_{FTA3\|E}$ |
| Tgt Type 4 | $P_{TR\ 4\|Type\ 1}$ | $P_{TR\ 4\|Type\ 2}$ | $P_{TR4\|Type\ 3}$ | $P_{TR\ 4\|Type\ 4}$ | $P_{FTA4\|E}$ |
| Non-Target | $1-\Sigma P_{TRj\ \|\ Type\ 1}$ | $1-\Sigma P_{TRj\ \|\ Type\ 2}$ | $1-\Sigma P_{TRj\ \|\ Type\ 3}$ | $1-\Sigma P_{TRj\ \|\ Type\ 4}$ | $1-\Sigma P_{FTAj\ \|\ E}$ |

The matrix could be further expanded by considering that the munition could encounter a variety of non-target types, each with its own column in the confusion matrix. Although this would add to the number of columns in the confusion matrix, it would not add to the number of rows, since the munition would not be type-specific in its declaration of a non-target.

Since $P_{TR}$ is the probability that an encountered target will be classified as a target of *any* type, it cannot be taken directly from the confusion matrix as it was for the single

target type case. For example, if a munition were to encounter a target of Type 1, the probability that it would classify the target as a target of any type is the sum of $P_{TR\ 1|Type\ 1}$, $P_{TR\ 2|Type\ 1}$, $P_{TR\ 3|Type\ 1}$, and $P_{TR\ 4|Type\ 1}$. To further complicate matters, the evaluation of equation 2.13 requires a composite estimate of $P_{TR}$ for all target types that may be encountered. This composite $P_{TR}$ can be obtained through a weighted average of $P_{TR}$ for each encountered target type. Let the probability that an encountered target of type i will be declared as a target of any type be defined as:

$$P_{TRi} = \sum_j P_{TRj|Type\_i} \tag{2.14}$$

where j ranges from one to the number of target types the ATR can recognize. Also, define the probability that an encountered target is of type i as:

$$P_{Ei} = \frac{\eta_{ti}}{\eta_{total\_tgts}} \tag{2.15}$$

A composite $P_{TR}$ weighted by the average densities of the various target types can then be defined as:

$$P_{TR} = \sum_i P_{TRi} \cdot P_{Ei} \tag{2.16}$$

**2.2.3 Outcome Trees.** In the search outcome tree for the single munition, single target case, there was only one path to success: find the one target and kill it. In the case of multiple targets, there are multiple paths to success, as shown by the solid lines in Figure 2.4 (6). If the munition encounters a target but does not recognize it, it simply continues to search and may still encounter another target. In spite of the changed search outcome tree, the outcome tree for engaging a declared target remains the same as Figure 2.3 (this will change for the multiple munition case).

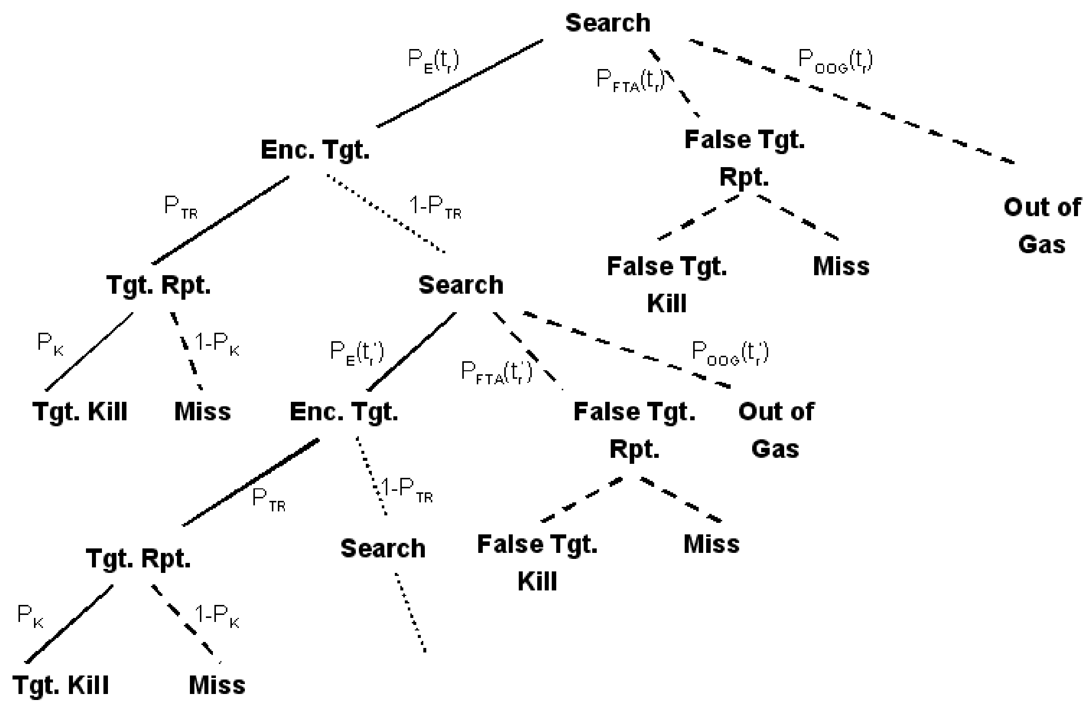



**Figure 2.4 Search Outcome Tree for Single Munition, Multi-Target Scenario**

Calculating the likelihood of a successful search in the same manner as for the single munition, single target case becomes problematic for the single munition, multiple target case. In the single target case, the probability of success was simply the product of the probabilities along the path to success. Applying the same logic to the multi-target case leads to an infinite series, since there are an infinite number of paths in the search outcome tree that lead to successful outcomes. Fortunately, this problem is alleviated by the method in which we developed $P_E(A_s)$. The nested characteristic of Figure 2.4 is due to the continual re-evaluation of $P_E$ at progressive time intervals. Since our definition for $P_E$ (equation 2.13) was integrally derived, the search outcome tree collapses into a single level, similar to the outcome tree for the single munition, single target case, and the probability of a successful search is similar to equation 2.8, albeit with a new $P_E$ (6):

$$P_{SS} = P_K \cdot P_{TR} \cdot \frac{\eta_t}{\eta_t P_{TR} + \alpha}(1 - e^{-(\eta_t P_{TR} + \alpha)\cdot V \cdot t_R \cdot W}) \tag{2.17}$$

Since the outcome tree for engaging a target did not change, the probability of a successful attack is identical to the single munition, single target case, as shown in equation 2.10.

## *2.3 The Multi-Munition, Multi-Target Case*

Even with multiple munitions, most of the theory developed for the single munition, multiple target case is applicable. In fact, the only difference between the two cases lies in the probability of a successful attack. This is due to the nature of the search and attack tasks. In every case presented in this research, the munitions essentially search independently. The likelihood of a particular munition having a successful search is dependent only on the characteristics of that particular munition—the characteristics or even presence of other munitions has no effect. The only time the multiple munitions have an effect is when one munition attempts an attack on a target declared by another munition.

The complexities associated with cooperative engagement is best shown through an outcome tree (see Figure 2.5):

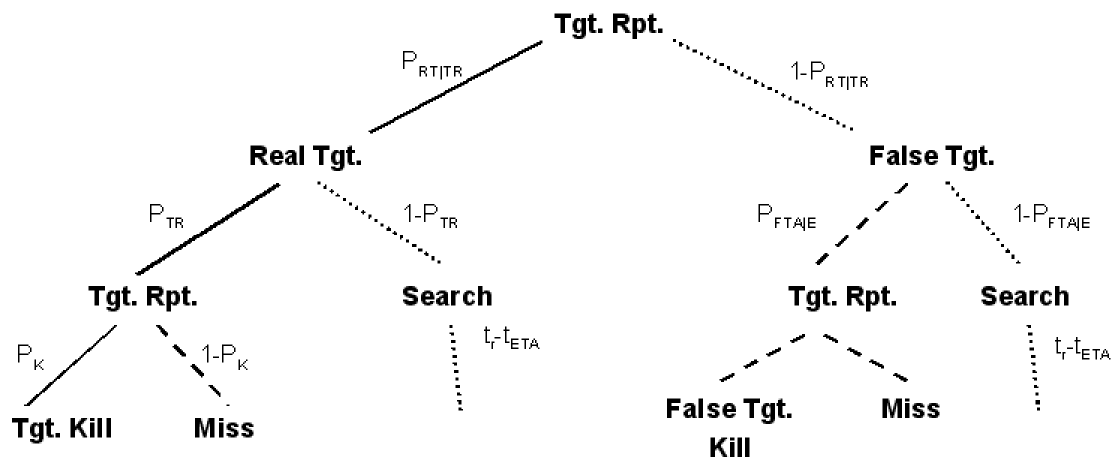



**Figure 2.5 Engagement Outcome Tree for Multiple Munition, Multiple Target Scenario**

Anytime a munition declares a target, there is a probability less than one that the declared target is in fact a real target. Whether or not the declared target is a real target, a munition that attempts to engage a target declared by another munition may or may not recognize the declared target for what it truly is, according to its own $P_{TR}$ or $P_{FTA|E}$. If the attacking munition also recognizes the declared target as a target, then it will execute an attack that may or may not be successful. If however, the attacking munition does not recognize the declared target as a target, it will return to searching, and may find another target or successfully attack a target declared by another munition. The probability of a successful engagement is simply the sum of the products of the probabilities along the paths that lead to positive outcomes, shown analytically in equation 2.18 (6):

$$P_{sa} = P_K \cdot P_{TR} \cdot P_{RT|TR} + P_{SS}(t_r - t_{ETA}) \cdot (1 - P_{TR}) \cdot P_{RT|TR} + \qquad (2.18)$$

$$P_{SS}(t_r - t_{ETA}) \cdot (1 - P_{FTA|FTE}) \cdot (1 - P_{RT|TR}) \quad .$$

The probability of a successful search is identical to the single munition, multiple target case since, as previously stated, the individual munitions search independently. The collapsible characteristic of the single munition, multiple target search outcome tree

is also present in the multiple munition, multiple target case. Thus, the probability of a successful search (and consequently the probability of a successful attack) can be calculated in closed form.

# III. The Computer Simulation

The MATLAB/Simulink simulation that was modified and used for this research was developed by AFRL/VACA as a developmental tool for their research in cooperative vehicles. The simulation itself was still in the development phase and was modified even as this research was being conducted, in part because of this research and in part due to other initiatives on behalf of VACA. This section discusses limitations of the initial simulation obtained from VACA and changes that were made to facilitate this research. Some of the changes discussed below were subsequently included in the VACA simulation, others were not.

## *3.1 Original Simulation*

**3.1.1 General.** The simulation as originally developed by VACA employs a maximum of eight vehicles searching for a maximum of ten targets and non-targets. The vehicles exhibit cooperative behavior in terms of target identification, target classification, and task allocation in order to improve mission effectiveness. A typical simulation begins with the vehicles starting from pre-determined positions and flying pre-determined routes. When an object enters a vehicle's field of regard, the vehicle classifies the object as a target or non-target and assigns a probability of correct classification based on the angle from which the vehicle viewed the object. Each vehicle then calculates the benefits of performing certain tasks. Possible tasks are: continue searching, re-classify (i.e. assist in classifying an object), attack, and perform battle damage assessment. Vehicle tasks are assigned such that the overall benefit is

maximized. This task allocation occurs each time the state of a target changes until the maximum simulation time is reached.

### 3.1.2 Task Allocation.

Task allocation for the various vehicles is modeled as a capacitated transshipment problem, described in detail in (10). The transshipment problem is a special case of linear programming and, if correctly formulated, yields an integer solution. A graphical representation of the network is shown in Figure 3.1 (12). In short, each vehicle is a supply node of capacity one. At the other end of the network is a demand of N, where N is the number of vehicles. The targets are transshipment nodes with supplies and demands of zero. Each vehicle therefore, must travel through the network to satisfy the end demand. The vehicles travel through the network along arcs that represent specific tasks and have certain benefits associated with them. The optimal task allocation provides the greatest overall benefit to the system.

The flow through the network is determined in part by the benefits associated with the various arcs. The values of these benefits are crucial to solving the network flow problem. Unfortunately, the formulas for calculating these benefits are widely varied. For the original simulation, a heuristic approach was used wherein emphasis was placed on killing high value targets over performing other tasks such as reclassification or battle damage assessment.

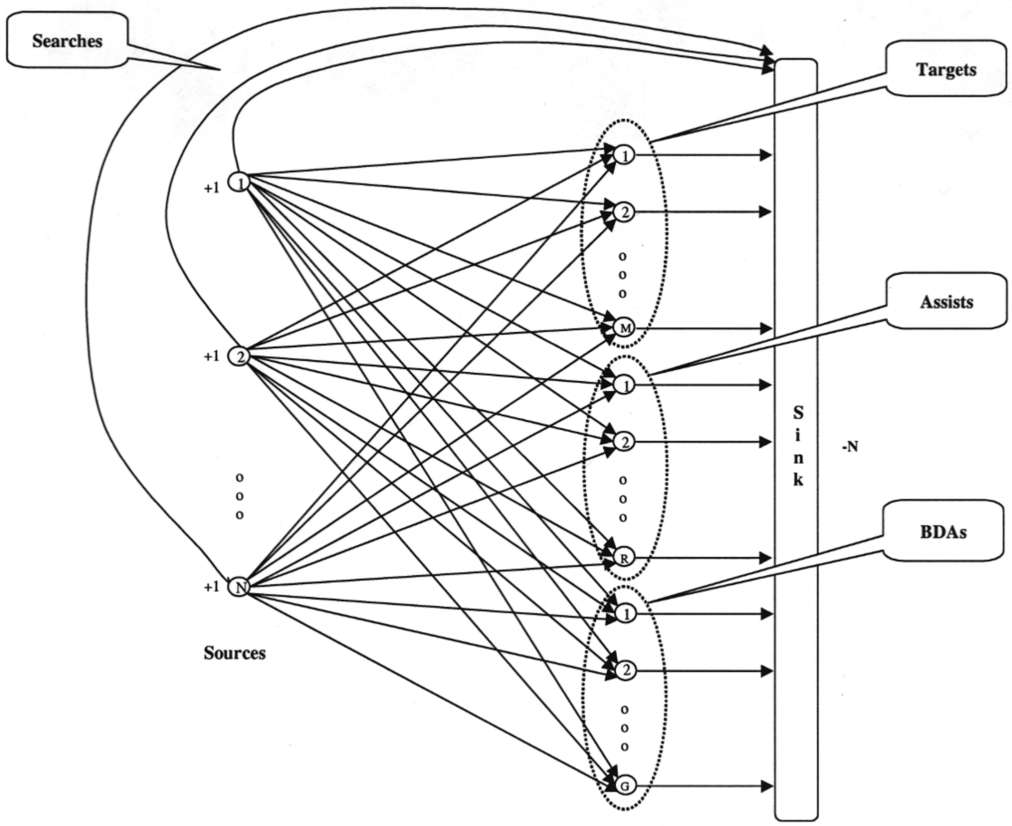



**Figure 3.1 Network Flow Model for Task Allocation**

**3.1.3 Target Classification and Confidence.** The original simulation does not provide any means for a vehicle to incorrectly identify an object. When a vehicle encounters an object, the vehicle always accurately classifies the object. Thus, there are no false targets. Instead of the possibility of false identification, the simulation uses a confidence level for correct classification. When a vehicle classifies an object, it calculates a confidence level for that classification based on the angle from which the vehicle viewed the object. If the confidence is below a pre-established threshold, then another vehicle may be (and usually is, depending on the results of the task allocation) assigned to assist in classifying the object. The second vehicle flies to the object and assigns its own confidence of correct classification, just as the original vehicle did. The

individual confidences are combined into a single confidence level that is compared to the threshold value. Once the confidence of correct classification is greater than the threshold, the object is deemed classified.

The classification scheme used by the simulation presents some difficulties. First is the fact that a vehicle cannot misidentify an object. According to Jacques (7) and Washburn (14), the possibility of misidentifying and subsequently attacking non-targets is crucial to evaluating the performance of cooperative search munitions. While it is certainly desirable to have vehicles with sophisticated ATRs that minimize misidentification, a perfect ATR is simply unrealistic. In order to accurately model real-life, the ATR model must have some error associated with it. This error is represented by the off-diagonals in the confusion matrix, as discussed in sections 2.1.3 and 2.2.2.

The problem of misidentified targets carries over into the logic for combining multiple sightings of a target. In the initial simulation, all sightings of a particular target are combined pair-wise to find a composite confidence of correct classification. However, this assumes that all of the classifications are of the same target type. If the vehicles are allowed to misidentify targets, then the combinatorial logic in the original simulation breaks down. When multiple classifications yield different results, those results cannot be combined to form a composite confidence level. Rather, the individual classifications must be compared with the other classifications and only the confidence levels associated with similar classifications combined into composite confidence levels. The classification with the highest composite confidence level can then be accepted as the most likely classification.

#### 3.1.4 Lethality.

In addition to the fact that the vehicles cannot misidentify objects, they also cannot miss a target during an attack, and all attacks result in a kill. Modifications were made to allow for non-lethal attacks (see section 3.2.4).

#### 3.1.5 Battle Damage Assessment.

One of the possible tasks that a vehicle may perform is battle damage assessment (BDA). After a target has been attacked, another vehicle may be assigned to assess the damage done to that target. In the original simulation, an attack always kills a target, and a battle damage assessment always confirms that kill—the BDA sensor is perfect. This is obviously not accurate to real-life and was modified as discussed in section 3.2.5.

#### 3.1.6 Communications.

The simulation entails perfect global communications. All of the vehicles are privy to, and make decisions based on, the same information. This makes cooperative behavior appear more beneficial than it truly is. In actual combat, the communications range would be limited and information may be mis-communicated or not communicated at all. In such a scenario, different vehicles would calculate benefits based on different information, and the network flow problem would produce sub-optimal results. While communications is certainly an important aspect of cooperative behavior, it was only partly addressed in this research. Modifications were made which allow the vehicles to communicate bad information (i.e. false targets), but the communications remained global and reliable (i.e. no miscommunication). Compared to other areas such as the autonomous target recognition model and lethality of the vehicles, the remaining communications issues seemed a relatively minor issue. They are addressed only as recommendations for future research.

## 3.2 Simulation Modifications

Several modifications were made to the simulation to accommodate the desired research:

- Increase the maximum number of targets
- Separate truth information from sensed information
- Provide a realistic ATR model
- Provide for non-lethal attacks
- Modify the battle damage assessment algorithm
- Change the frequency of the optimization (task allocation) calculations
- Modify the benefit calculations used in the task allocation routine
- Other modifications

These modifications are described below. The resulting simulation, with all of these modifications, was used to compare the results of cooperative search and engagement with non-cooperative search and engagement.

**3.2.1 Maximum Number of Targets.** The first step in adapting the simulation for this research was increasing the number of targets. While conceptually this may seem to be a simple task, it was actually quite complex. Much of the Simulink layout in the original code used individual paths for signals pertaining to each target. The individual signals were obtained by de-muxing outputs from preceding routines. This allowed each individual value in the routine's output to be named, arguably improving traceability in the system. However, in order to increase the number of targets, new paths had to be added for each new target—a task that quickly becomes tedious.

Instead of de-muxing the output signals into many paths of individual signals, the output signals were de-muxed into fewer signals of larger size. For example, in the original code, a routine which output the cost of a vehicle attacking the various targets and the estimated time for that vehicle to attack the various targets would have an output vector of size $(2 \cdot N_{tgts})$. That output vector would be de-muxed into $(2 \cdot N_{tgts})$ individual signals. In the modified code, the output vector is de-muxed into only 2 signals, each of size $N_{tgts}$. With this modification, the number of targets can be increased or decreased with relative ease since many of the requisite changes in Simulink will occur automatically.

**3.2.2 Truth Information vs. Sensed Information.** The original simulation did not distinguish between truth information and sensed information. This led to dilemmas in several places throughout the simulation. For instance, when a vehicle encountered an object, it would classify the object according to truth information. Likewise, when a target was attacked, the remaining vehicles would recognize the target state as the true target state. Such logic does not accurately represent real world scenarios. To bring the simulation closer to reality, new variable structures were created to keep track of information as sensed by the vehicles. The accuracy of the sensed information is dependent on probability matrices that were added to the logic and discussed below.

The separation of truth and sensed information has significant ramifications. Previously, the vehicles allocated tasks based on truth information. Now, decisions are made based on information which may or may not be accurate. Thus some task allocations may not maximize the benefit (and may even be detrimental) in reality. This

incorporates some of the possible disadvantages of cooperative behavior. For non-cooperating vehicles, bad information will only affect the vehicle incorrectly sensing the information. With cooperating vehicles, the bad information is passed to all of the vehicles and propagated through the entire network.

**3.2.3 Autonomous Target Recognition.** The original code did not provide logic for a realistic autonomous target recognition model. With the original code, a vehicle always classified an object as what it truly was (target or non-target). The decision to attack or not was based on an estimation of the quality of the classification (labeled the ATR metric), which in turn was based on the angle from which the vehicle observed the object. A vehicle would not be assigned to attack a target until the ATR metric was greater than an established threshold. This setup was deemed inadequate for this research.

New logic was added to allow the vehicles to misidentify objects. This was accomplished via a confusion matrix (see sections 2.1.3 and 2.2.2). When a vehicle encounters an object, a call is made to a function referencing the confusion matrix. Depending on the true type of the object encountered, the vehicle will classify it based on a random draw and the probabilities in the confusion matrix. This provides a significant improvement to the simulation. For one, it allows vehicles to misidentify objects, thus allowing false target attacks—something the original code never permitted. The use of a confusion matrix also allows different probabilities of classification for different object types. So, if a particular non-target happens to look similar to an actual target, the confusion matrix would have a relatively high probability that the non-target might be classified as the target, or vice-versa.

#### 3.2.4 Non-Lethal Attacks.

Non-lethal attacks fall into two categories: missed targets and low lethality warheads. For the purposes of this research, the two categories are combined and simply called non-lethal attacks. In the original simulation non-lethal attacks were not possible, although a pre-determined number of attacks could be required before considering a target killed. This logic was modified to allow a hit to be non-lethal. This was accomplished with a random draw and probability of kill ($P_k$). Thus, when a vehicle attacks a target, a random draw is made and compared to the value of $P_k$. If the random draw and $P_k$ value are such that a successful hit occurs, then the target is considered killed. However, this information is not communicated to the other vehicles. The vehicle that would communicate that information—the attacking vehicle—is also dead and unable to provide that information. Thus, the non-attacking vehicles know that an attack was made on a target, but they do not know if the attack was successful.

#### 3.2.5 Battle Damage Assessment.

BDA has the potential of significantly widening the scope of a simulation as issues such as heterogeneous vehicle types or separate sensors within a single vehicle become significant. In an attempt to limit the scope of this research, the BDA task was eliminated. The benefit of the BDA task was set to zero, so it is never advantageous for a vehicle to perform BDA. Rather, vehicles only track the number of times a particular target has been attacked and incorporate a probability that the target is still alive into the benefit calculations (discussed in section 3.2.7).

#### 3.2.6 Task Allocation.

As previously stated, the optimization routine for allocating vehicle tasks was event triggered in the original simulation. Under certain

conditions, this scheme had undesirable results. For example, if a low priority target was encountered early in the simulation, the optimization routine may have determined that it was better for all of the vehicles to continue searching in the hopes of finding a high priority target. If no other targets were found, then the optimization routine would never be reactivated, benefits would not be recalculated, and tasks would not be reassigned. Thus, the vehicles would continue to search until they ran out of gas instead of searching for a period of time and then attacking the known low priority target. To avoid this situation, a modification was made to trigger the optimization routine if a certain time period had passed without the optimization routine being triggered by an event. Thus, in the above example, the optimization routine would first be triggered by the low priority target encounter and subsequently triggered at regular time intervals.

**3.2.7 Benefit Calculations.** The benefit calculations used in the original simulation were based on heuristics which, although intuitively appropriate, did not have strong mathematical support. For this research, a more rigorous method was developed and implemented based on the probabilities of success discussed in Chapter II.

The development of the search benefit was relatively straight forward, and the final result is shown in equation 3.1.

$$\textit{Search Benefit} = \xi \cdot P_{ss} \tag{3.1}$$

In this equation, a weighting parameter, $\xi$, is applied to the probability of a successful search developed in Chapter II (equation 2.17). The weighting parameter $\xi$ is the relative merit of searching for additional targets versus attacking a known target and can vary within the range 0 to 1. Thus, if $\xi$ is 0, there is no benefit obtained from continued

searching, while if $\xi$ is 1, emphasis is placed on continued searching rather than attacking targets.

The benefit calculation for attacking a target is more complex, and took on different forms depending on the sensed target priority.

High Priority Target: $$Attack\ Benefit = (1-\xi)\cdot(\frac{t_R - ETA}{t_R})\cdot(1-P_k)^n \cdot P_{sa} \quad (3.2)$$

Low Priority Target: $$Attack\ Benefit = (1-\xi)\cdot\beta\cdot(\frac{t_R - ETA}{t_R})\cdot(1-P_k)^n \cdot P_{sa} \quad (3.3)$$

Non-Target: $$Attack\ Benefit = 0 \quad (3.4)$$

Equations 3.2 through 3.4 not only consider the probability of a successful attack (equation 2.18), but also the probability that the target is alive and the estimated time to reach the target. The probability that the target is alive is obtained by a simple binomial probability calculation:

$$Probability\ target\ is\ alive = (1 - P_k)^n \quad (3.5)$$

where $P_k$ is the probability that an attack on a target will result in a kill, and $n$ is the number of attacks that have been made on the target. Thus, as more attacks are made on a target, the likelihood of the target still being alive decreases, discouraging subsequent attacks. This is an improvement over previous work by Gillen (3). In Gillen's research, more than two attacks on the same target was discouraged regardless of the warhead lethality. With equation 3.5, the warhead lethality is considered in the probability calculation. Thus, more attacks will be allowed for low lethality warheads and fewer attacks will be allowed for high lethality warheads.

The estimated time to reach the target is also important in the attack benefit calculation. The logic is: if the time to get to the target is greater than the time remaining

in the simulation (i.e. the time remaining until the vehicle runs out of fuel), then there is no point in trying to attack the target. However, if the vehicle can get to the target before running out of fuel, then an attack may be warranted. The mathematical translation is the term

$$\frac{t_R - ETA}{t_R} \tag{3.6}$$

Unfortunately, this particular term had other effects that were detrimental to the system's performance, as discussed in section 5.5.1. In subsequent revisions, the term was removed from the benefit calculations and logic added to prevent munitions from attempting to attack targets that could not be reached.

Finally, weighting parameters were applied to the attack benefits. The weighting parameter for attacking a target rather than continuing to search is simply the complement of $\xi$. For low priority targets, a second weighting factor $\beta$ was added to represent the value of low priority targets relative to the value of high priority targets. If low priority targets were just as valuable as high priority targets, then $\beta$ would equal 1. Likewise, if low priority targets were not important at all, $\beta$ would equal 0.

In some of the scenarios explored in this research, the vehicles were allowed to cooperatively classify objects in an attempt to reduce the false target attack rate. For these scenarios, another benefit calculation was needed for re-classifying objects. The benefit for this task was made identical to the attack benefit. With this scheme, a vehicle will not attempt to re-classify a target unless initial classification attempts indicate that the object may be a target worth attacking. Thus, search time is not wasted trying to make sure a non-target really is a non-target.

#### 3.2.8 Additional Modifications.

In order to facilitate this research, several other modifications were made to the original simulation. These modifications allow summary and output to a file of specified statistics, allow the activation or de-activation of various features (such as cooperation), and facilitate the optimization of the benefit calculations via response surface methodology. While these changes were important for this research effort, they did not affect how the actual simulation ran and are not described in detail here.

# IV. Response Surface Methodology

## 4.1 Introduction

Response surface methodology (RSM) is a process where statistical and mathematical design techniques are combined with empirical model building to systematically determine a process's response with respect to certain inputs. The technique has been widely used in industrial applications and has also seen use in developmental applications as a means to improve or optimize processes and products. The goal in RSM is to determine optimum parameter settings by accurately mapping the process response, called the response surface, in as few experimental runs as possible, thereby saving time and money.

RSM can be broken into several phases, as described by Myers and Montgomery (9). The first phase, Phase 0, is a screening experiment. In this phase, several factors that may affect the response are identified and tested to determine the significance of each. This is accomplished by conducting experiments (in this case simulation runs) with the factors set at various levels and recording the resulting responses. A regression model is fit to the data using traditional linear regression techniques (see reference 10 for a review of linear regression methods), and the significance of individual coefficients in that regression model tested via a statistical t-test. Those factors that provide little insight or influence on the response surface are discarded, and the most important factors are kept. This phase is crucial to reducing the number of subsequent runs and becomes more important as the number of possibly influential factors increases.

In Phase 1, an attempt is made to determine the vicinity of the optimum point. A regression model is created using those significant factors found in Phase 0. If this model indicates that the optimum point is far from the current design region, then the experimenter moves the design region closer to the estimated optimum point via a variety of optimization techniques (i.e. the method of steepest ascent) and starts the process over by conducting new Phase 0 experiments. Once the design region is near the estimated optimum point, Phase 2 begins.

The goal in Phase 2 is to more accurately determine the location and nature of the optimum point. This is accomplished by constructing higher order models of the response surface. Additional experiments (or simulation runs) are usually required for this phase so that the precise nature of the curvature can be determined. Once the nature of the curvature is estimated, a final optimum point may be settled on.

One of the advantages of RSM over other optimization techniques lies in its sequential process. If designed correctly, the results from each phase can be used for the next phase, thereby minimizing the total number of runs. It is also quite adaptable to different testing environments. If testing resources are abundant, then relatively loose criteria may be used in Phase 0 to determine significant factors, resulting in more factors being kept in the model and perhaps increasing the fidelity of the subsequent models. However, if testing resources are limited, then strict criteria may be used and the number of significant factors, along with the requisite number of runs, greatly reduced. RSM also provides more insight into the nature of the response surface. Instead of simply determining an optimum point, the method provides gradient information near the

optimum point. This allows the designer to balance the sensitivity of the optimum settings with the desire to achieve the best response.

## *4.2 Application*

RSM was used in this research to find the optimum settings for the two weighting parameters ($\xi$ and $\beta$) applied to the benefit calculations discussed in Chapter III. The relative value of each parameter by no means reflects the relative importance of the factors. While there were differences in the significance of the two factors, the optimum settings are just that: the factor settings that provide the optimum response—not an indicator of significance.

**4.2.1 Independent Variables.** Two independent variables were optimized: $\xi$ and $\beta$. Recall from section III that $\xi$ describes the relative merit between continuing to search for additional targets versus attacking a known target, and $\beta$ describes the value of low priority targets to high priority targets. Each factor independently varied (in natural variables) between 0 (the low setting) and 1 (the high setting).

**4.2.2 Responses.** Two responses were chosen to represent the mission effectiveness of each simulation run:

- Number of Targets Killed, and
- Hit Formula.

The number of targets killed was simply the total number of targets (of any priority) that were killed. The hit formula, taken from Gillen's research (3), is a mathematical formula that considers the different priority targets that were hit as well as hits on non-targets:

$$Hit\ Formula = 2 \bullet (\#\ high\ priority\ hits) + (\#\ low\ priority\ hits) - (\#\ non\text{-}target\ hits) \quad (4.1)$$

With this formula, more benefit is gained from attacking high priority targets versus low priority targets, and attacks on non-targets result in a penalty. Both responses were simultaneously optimized. Thus, the final optimum parameter settings may not maximize either response individually, but do represent the optimum balance of the two responses. Each response could be weighted to indicate their relative importance. For this research, each response was treated with equal importance.

#### 4.2.3 Phase 0 Screening Experiments.

Although only two factors were considered for the response surface, a screening experiment was performed. Because of the small number of factors, a full factorial design was chosen. A conceptual presentation of this design is shown in Figure 4.1, where each dot represents an experimental setting.

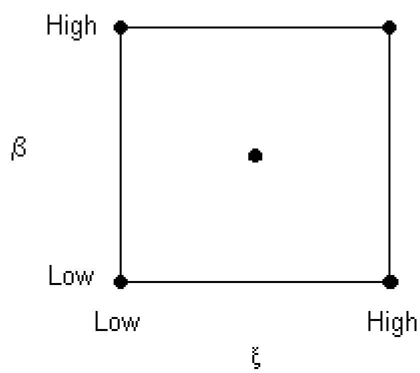



**Figure 4.1 Two Factor, Full Factorial Screening Design**

With the full factorial design, there would be no confounding of effects. Thus, higher order effects and interactions could be estimated in addition to the main effects. This design required a minimum of four runs (the four corners of Figure 4.1). Three replications were made at each design point and three center runs were included for a

total of 15 runs. The center runs allow testing for the presence of curvature (i.e. a quadratic model), though the exact nature of the curvature (i.e. whether it was with respect to $\xi$ or $\beta$) still requires additional runs. The 15 runs in the screening experiment were carried over into subsequent experiments (Phase 1) where they were augmented with additional runs as required (thus the screening experiments did not add any to the total number of runs required).

In most cases, the screening experiments indicated that both factors were significant (at a significance level of 0.10). However, in some scenarios, one of the parameters could be dropped and the experiment simplified. In such cases, the runs from the screening experiment could be used to construct a model (linear or quadratic) and determine the optimum settings without any additional runs.

Another important piece of information obtained from the screening experiments was the presence of curvature. Since replications and center points were included in the experiment, a lack of fit test could be performed. In most scenarios, the lack of fit test indicated the presence of curvature with regards to at least one of the responses.

**4.2.4 Phases 1 and 2.** For this study, Phases 1 and 2 were essentially combined. This was done for two reasons. First, the screening experiments typically indicated the presence of curvature, thus using a linear model for a Phase 1 experiment (as is often done) did not seem reasonable. In addition, the software used for constructing the models could construct second order models just as easily as first order models. The second reason for combining Phases 1 and 2 was the range of the weighting parameters. As previously stated, each parameter could vary between 0 and 1. This range was small enough that the entire range could be used for the experiment without

major fears of missing optimum points (due to too coarse of an experiment). Since the entire range of the parameters was used in the experiment, the design region could not be moved to bring it closer to an external optimum point. Thus the initial design region would also be the final design region, and a higher order model was constructed.

The design chosen for this phase was the central composite design (CCD), shown graphically in Figure 4.2.

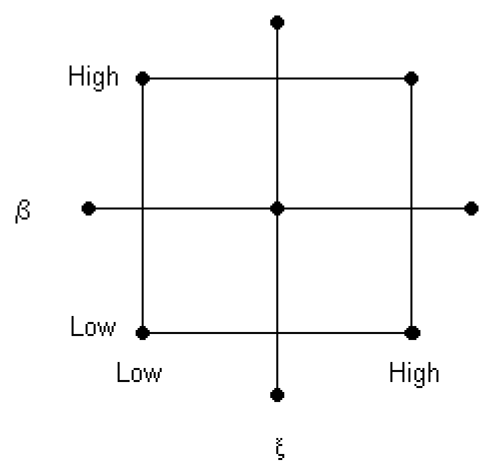



**Figure 4.2 Two Factor, Central Composite Design**

According to Myers and Montgomery (9), the general CCD is composed of three parts:

- A $2^k$ factorial or fraction of resolution V or greater. This portion of the CCD is required for the estimation of the linear terms and is the only portion that contributes to the estimation of interaction terms. It is orthogonal, resulting in the minimum variance possible for the number of runs performed.

- Axial points. With axial points, one factor at a time is varied while the other factors are held at their mid-level values. The axial points contribute to the estimation of individual quadratic terms, but not interactions.
- Center runs. Center runs allow the sum of the quadratic terms to be estimated, providing an estimation of whether curvature is present or not. They do not however, allow individual quadratic terms to be estimated. They also serve to stabilize the design (especially important with spherical designs).

There are several nice aspects to the CCD and its application to this research. The Phase 0 screening experiment consisted of a full $2^2$ factorial with three replicates. This served as the first portion of the Phase 2 CCD and also fulfilled the requirement for center points (Myers and Montgomery recommend three to five center points). However, a quirk in the data reduction software used for this research required additional center runs. Using the data reduction software (JMP 4.0.4), an orthogonal CCD with three replications was set up. The software required a minimum of eight center runs for this design. Thus, while the three center runs from the screening experiment could be applied to the CCD, an additional five runs had to be completed. This reduced the efficiency (in terms of number of runs) of the RSM, but also served to increase the stability of the design by reducing the variance. This trade-off was deemed acceptable, especially since the axial points created a spherical design, as described below.

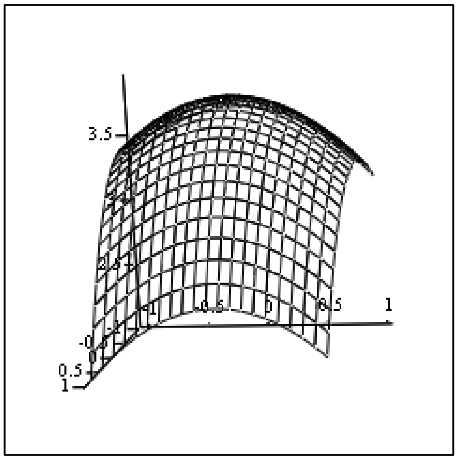
The axial distance for the axial point runs was chosen so that the resulting design would be both orthogonal and rotatable. An orthogonal design is variance optimal, providing the least amount of variance compared to any other design with the same

number of runs. Rotatability is desirable for prediction purposes. With a rotatable design, variance is only a function of distance from the center of the design region. This allows for consistent estimates of the response across the entire design region. The axial distance required for both orthogonality and rotatability was $\sqrt{2}$. The downside of this axial distance was that it created a spherical design, which is a singular design when used to fit a second order model. Center points alleviate the singularity, and since an abundance of center points were already built into the design, the singularity of the design was not a concern.

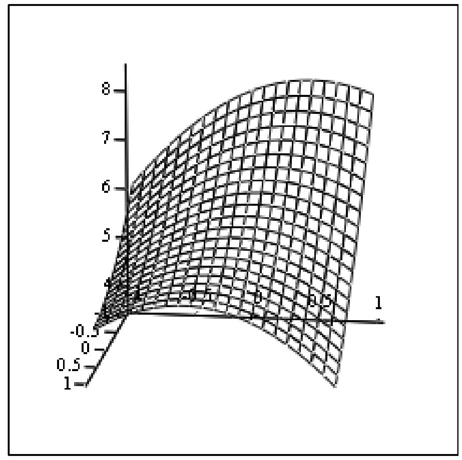
A total of 17 additional runs were required to complete the CCD. The majority of these runs (12) were axial runs with replicates. The other five runs were center runs required by the data reduction software.

As previously stated, two responses were simultaneously optimized: the total number of targets killed and the target formula. The goal was to maximize both responses. This was accomplished through the use of desirability functions, which were a feature of the data reduction software. With this method, each response is given a weight relative to the other and the individual responses are essentially combined into a single response. The single response is then maximized. This can be (and was for several of the experiments, as a check) accomplished by hand, as follows:

- Models for each response were constructed from the CCD. Since the responses were functions of only two variables, the responses could be plotted in three dimensions, as shown in Figure 4.3.




TgtsKilled



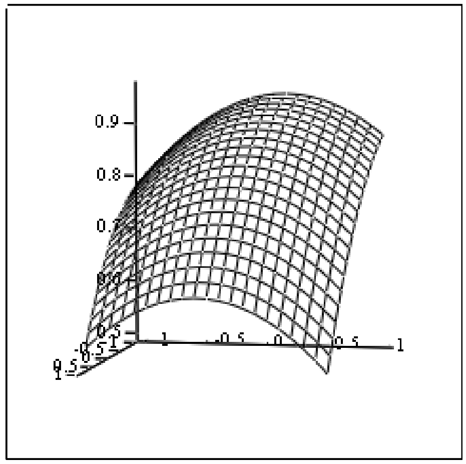

Formula

**Figure 4.3 Sample Response Plots**

- The responses were then non-dimensionalized by dividing each value by the maximum value for that particular response. These non-dimensional responses were then averaged according to equation 4.2:

$$Composite_{i,j} = \frac{\dfrac{TgtsKilled_{i,j}}{\max(TgtsKilled)} + \dfrac{Formula_{i,j}}{\max(Formula)}}{2} \qquad (4.2)$$

  In equation 4.2, the responses are equally weighted. If one desired to give priority to one response over the other, then coefficients indicating the relative importance could simply be applied to the non-dimensional responses. A sample plot of the combined responses is shown in Figure 4.4:




Composite

**Figure 4.4 Sample Plot of Non-Dimensional Composite Response**

- The optimum settings of the input variables could then be found by finding the maximum of the composite response. For those experiments where this hand-optimization was accomplished, results matched those obtained by the data reduction software.

# V. Results and Analysis

The results presented in this chapter can be broken into two phases. The first phase consists of the results from the processes and experiments described in the previous chapters of this thesis. These results are presented in sections 5.1 through 5.4. The results, while informational, were somewhat unexpected. Specifically, cooperative behavior performed worse than non-cooperative behavior. This led to the second phase, in which modifications were made to both the behavior algorithm and the optimization process in an attempt to achieve more desirable results. The changes and the improved results are presented in section 5.5.

## *5.1 Response Surface Methodology*

**5.1.1 Weighting Parameters.** As previously discussed in Chapter III, weighting parameters were applied to the benefit calculations. The weighting parameter $\xi$ represents the relative weight of continuing to search for targets versus attacking a declared target, and $\beta$ represents the value of low priority targets relative to high priority targets. These weighting parameters were optimized for the various scenarios using RSM. The results obtained from the RSM are interesting and bear mention here.

For all of the scenarios examined, $\xi$ is statistically more significant than $\beta$. In approximately half of the cases, $\beta$ is not statistically significant at all (at the 0.1 significance level), while $\xi$ is significant in every case. However, the level of significance of the two weighting parameters does not appear to follow any trends. Their significances vary between scenarios and between cases within given scenarios.

In all of the scenarios examined, the optimum setting for $\xi$ is zero. This can be considered a desirable characteristic. The lack of variability between scenarios means that the scenario parameters do not influence $\xi$. Thus, $\xi$ does not need to be adjusted for a particular scenario—it is quite robust in that sense. Unfortunately, this result is likely due to the choice of responses optimized in the RSM. By using the hit formula (equation 4.1) as one of the responses, too great of an emphasis was placed on attacking targets versus actually killing targets. Thus, the optimum setting for $\xi$ always promoted attacking known targets rather than searching for additional targets. The optimum settings for $\beta$ are more variable. With respect to this weighting parameter, the RSM at times yielded optimum points at both extremities of the allowable range (i.e. 1 or 0), depending on the particular case. Again, there did not appear to be any link between which setting was optimum and the particular scenarios.

## 5.2 Number of Targets Killed

Three cooperation algorithms were examined: no cooperation, cooperative engagement, and cooperative classification and engagement. The effects of the different algorithms are presented in this section.

Contrary to expectations, cooperative logic did not increase the number of targets killed (see Figure 5.1). For low lethality warheads ($P_k$ = 0.5), there is no statistical difference between the three cooperation algorithms examined (at a 95% family confidence level). This seems to be due to the cooperative logic and the characteristics of the warhead itself. If we look at the number of targets attacked (Figure 5.2) and the number of attacks made on those targets (Figure 5.3), we see that the cooperative cases made many attacks on just a few targets, while the non-cooperative cases attacked more

targets, but made fewer attacks overall. The non-cooperative cases suggest that the warhead often was not lethal enough to kill a target, even with up to two attacks on a given target (since the munitions had a 50% overlap in their coverage). The fact that the cooperative cases did not fair any better indicates a possible disadvantage of cooperative behavior. With these cases, the munitions attacked a fewer number of targets, automatically limiting the number of targets that could possibly be killed. If some of the munitions had searched longer rather than attacking a target that had been attacked multiple times, the results may have been better.

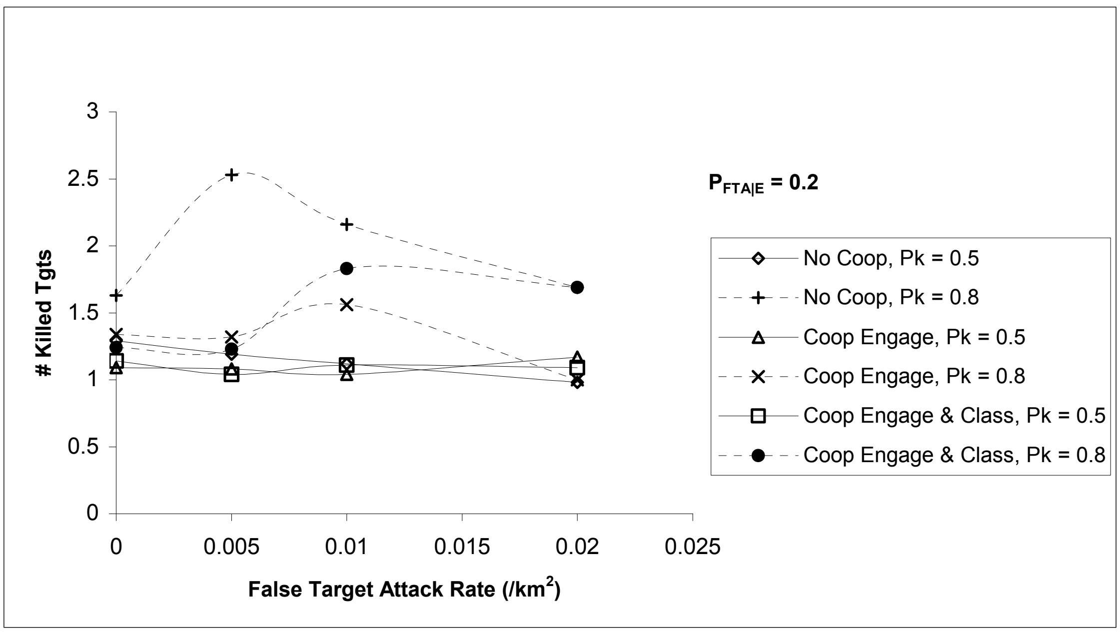



**Figure 5.1 Number of Killed Targets From Initial Research**

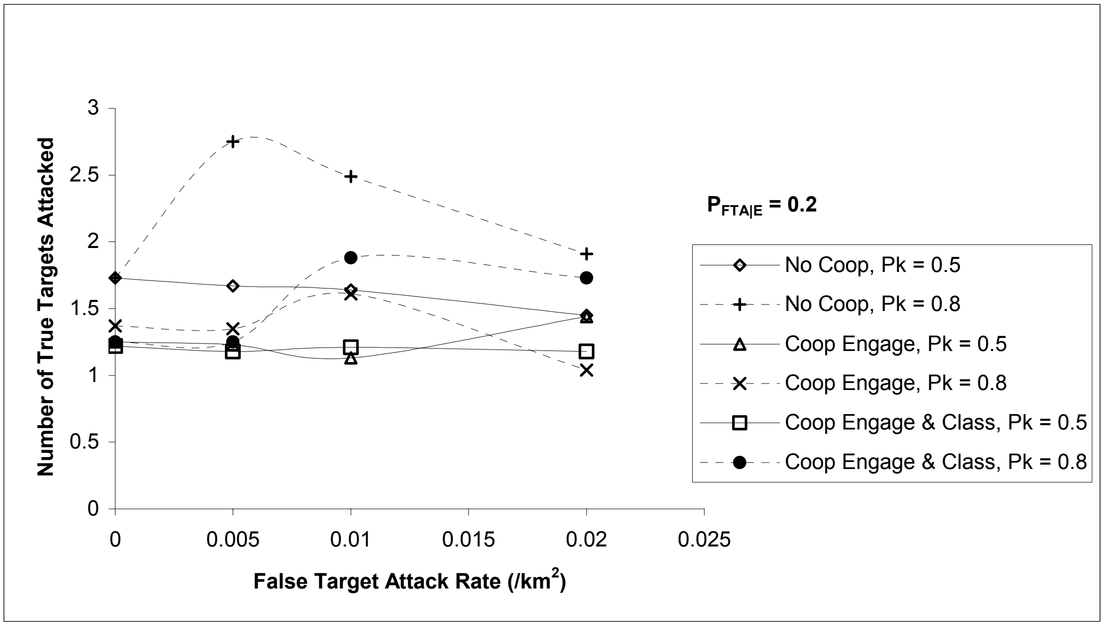



**Figure 5.2 Number of Targets Attacked From Initial Research**

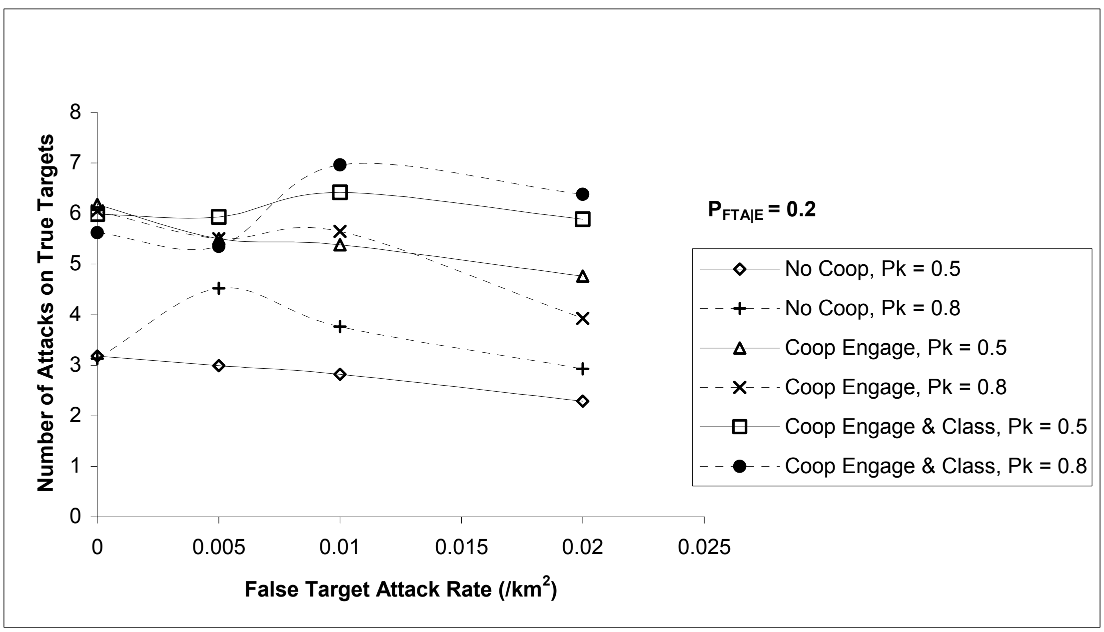



**Figure 5.3 Number of Attacks on Targets From Initial Research**

Conclusions for the high lethality cases ($P_k = 0.8$) are somewhat more difficult to make. In these cases, the non-cooperating munitions perform better than either of the cooperative algorithms. Again, this is likely due to the cooperative munitions executing

many attacks on only a few targets (see Figure 5.2 and Figure 5.3). The effects of this behavior are so dominant at low false target attack rates that the high lethality cases with cooperation do no better than the low lethality cases. The high lethality cases also exhibit interesting behavior with respect to the false target attack rate. At false target attack rates of approximately 0.005 to 0.01, there is an increase in the number of targets killed. The consistency of this behavior across multiple scenarios suggests a real trend, though the reason behind this behavior is currently unknown.

## 5.3 Hit Formula.

Although cooperative behavior did not increase the number of targets killed, it did improve the quality of the targets attacked, as can be seen in an analysis of the hit formula response (Figure 5.4). The cases with cooperative behavior consistently outperform the cases without cooperative behavior. This is due to the fact that cooperative behavior allows many munitions to be brought to bear against high priority targets. However, cooperative behavior alone does not solve all of the problems encountered with non-cooperating munitions. As the false target attack rate increases, both the non-cooperative cases and the cases with only cooperative engagement suffer similar performance degradation. Even though the munitions in the cooperative engagement cases attack targets seen by other munitions, thereby increasing the likelihood of killing the target, they still rely on a single munition to identify the target. Thus, any deficiencies in the individual munition's ATR can become compounded. This can be solved in part through the use of cooperative classification. Notice in Figure 5.4 that for scenarios using cooperative engagement *and* classification, the rate of decrease in the value of the hit formula with respect to the false target attack rate is much less than

for either cooperatively engaging munitions or non-cooperating munitions. This reduction in the effective false target attack rate is discussed in more detail in section 5.4.

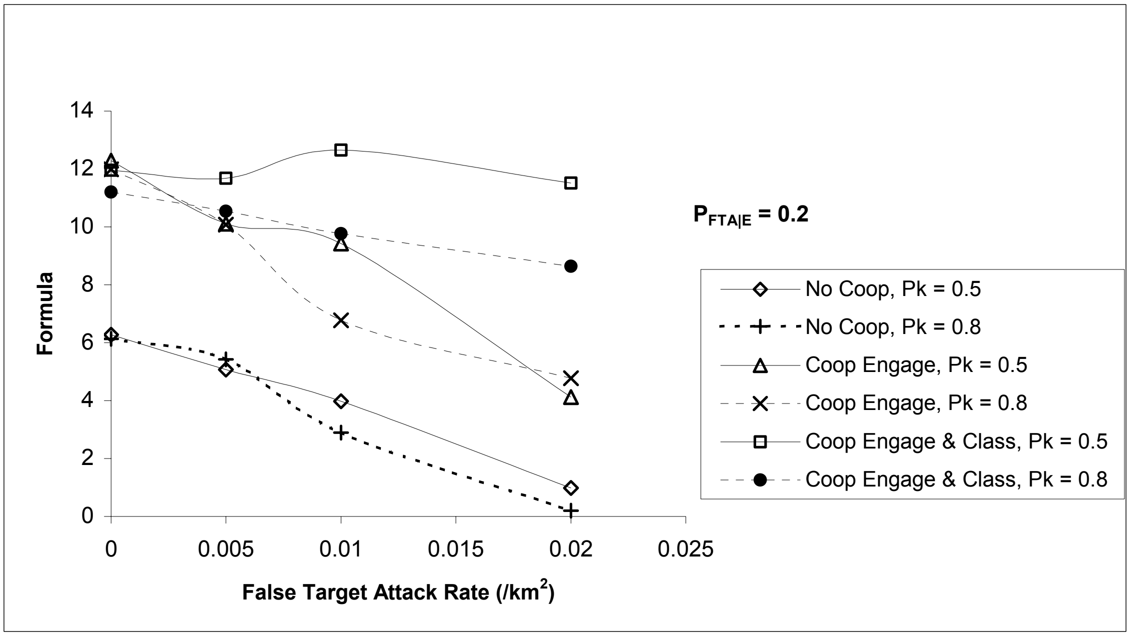



**Figure 5.4 Hit Formula Results From Initial Research**

For each cooperation algorithm, it appears that the low lethality warhead ($P_k$ = 0.5) outperforms the high lethality warhead ($P_k$ = 0.8). This is due to the form of the attack benefit calculation (equations 3.2 and 3.3). In calculating the benefit of attacking a target, an individual munition considers the probability that the target is still alive, even if attacks have already been made against that target (equation 3.5). Low lethality warheads will require more attacks on a given target before the target is likely to be dead. Thus, the results in Figure 5.4 are reasonable.

## *5.4 False Target Attack Rate*

The false target attack rate ($\alpha$) is important to any discussion of weapon effectiveness. With the implementation of cooperative behavior in autonomous weapons,

the subject takes on even greater significance. Cooperative behavior has great potential to improve the performance of weapon systems, but there are many potential pitfalls that must be addressed. One such pitfall is the communication of bad information. If a munition misidentifies a non-target as a target, that error is propagated through the entire network of weapons with results that may be worse than if the munitions did not communicate at all. A proposed solution to this problem is to have the munitions cooperatively classify potential targets, but this may have the undesirable effect of forcing munitions to spend excessive time trying to classify a single object. This research investigated these concerns by tallying the number of false target attacks in any given scenario. Conceptually, if cooperative behavior and the propagation of bad information significantly affect mission success, the number of false target attacks should be higher for cooperative engagement than for no cooperation at all. Likewise, if cooperative classification can alleviate the effects of bad information, the number of false target attacks should be lower. The results are presented below.

Figure 5.5 shows the effects the various cooperation algorithms had on the number of false target attacks. First, notice the similarity between results obtained from the no cooperation algorithm and the cooperative engagement algorithm. For the most part, the algorithms yielded results that were not statistically different (only the non-cooperative, high lethality case had any points that were significantly different at the 0.5 significance level). Thus, it appears that the propagation of bad information in a cooperative environment may not significantly increase the false target attack rate.

Notice however, the significant improvement gained with the use of cooperative classification. Both the cooperative engagement cases and the no cooperation cases

waste increasingly more munitions as the false target attack rate increases. With cooperative classification however, the number of false target attacks remains low. Cooperative classification appears to effectively lower the false target attack rate. This conclusion is significant. Results such as these mean that shortcomings in a munition's ATR may be overcome in large part by group behavior.

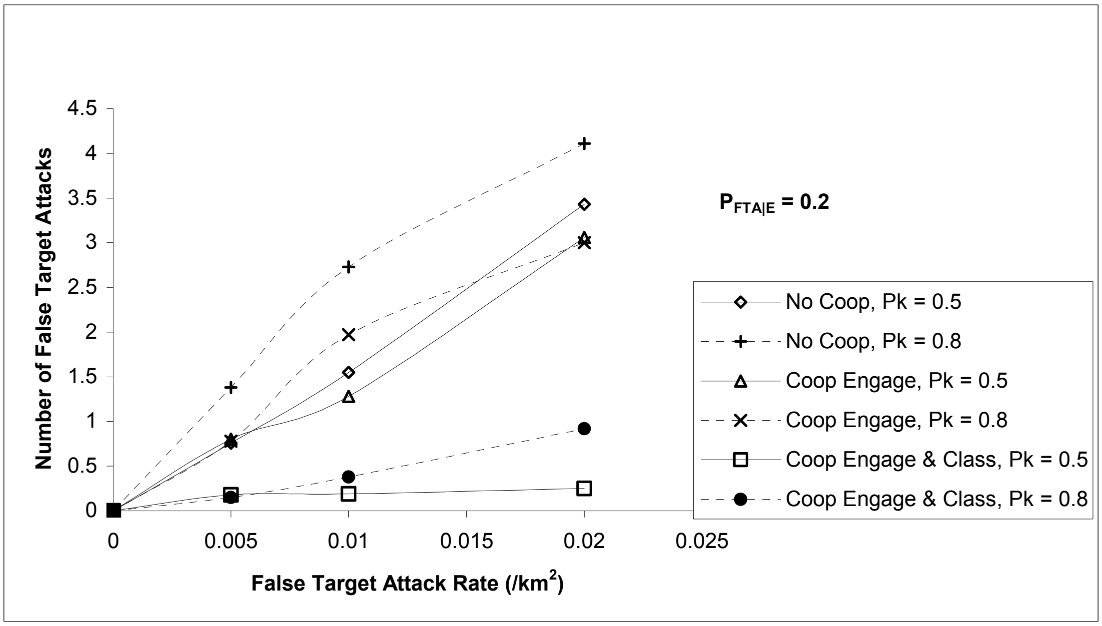



**Figure 5.5 Number of False Target Attacks From Initial Research**

## *5.5 Exploratory Excursions*

As previously mentioned, the initial results obtained in this research and discussed above did not produce the desired results. While there may indeed be scenarios where cooperative behavior leads to poorer results than non-cooperative behavior, it was thought that the degradation with cooperative behavior seen in this research was due to other factors. This led to a re-examination of the process and metrics used for this research. Several areas were modified, and limited exploratory experiments conducted. Those modifications and the ensuing results are presented in this section.

**5.5.1 Benefit Calculations.** The basic problem with the initial results was that too many munitions were attacking a single target too early in the simulation. This was in large part a result of the benefit calculations themselves. Several deficiencies were identified in the development and application of the benefit calculations. They are addressed below.

In its original form, the calculation of the benefit of attacking a target (equations 3.2 and 3.3) was a function of the probability of a successful attack ($P_{sa}$) (equation 2.18) which in turn was a function of the probability of a successful search ($P_{ss}$) (equation 2.17). As the time remaining in the simulation decreased, $P_{ss}$ decreased. This caused $P_{sa}$ to decrease, though not as quickly as $P_{ss}$. This was due to the fact that $P_{sa}$ had a non-zero base value while $P_{ss}$ did not (compare equations 2.17 and 2.18). Thus, $P_{ss}$ prematurely became less than $P_{sa}$. This effect carried into the benefit calculations and forced munitions to attack targets when it would have been more appropriate for them to continue searching. The problem was alleviated by effectively decreasing $P_{ss}$ as it is used in calculating $P_{sa}$. A multiplier of 0.01 was applied to the appropriate components of equation 2.18 so that the $P_{sa}$ became:

$$P_{sa} = P_K \cdot P_{TR} \cdot P_{RT|TR} + 0.01 \cdot P_{SS}(t_r - t_{ETA}) \cdot (1 - P_{TR}) \cdot P_{RT|TR} + \quad (5.1)$$

$$0.01 \cdot P_{SS}(t_r - t_{ETA}) \cdot (1 - P_{FTA|FTE}) \cdot (1 - P_{RT|TR}) \quad .$$

With equation 5.1, $P_{sa}$ becomes more constant and remains less than $P_{ss}$ for a longer period of time.

Equation 2.18 (and subsequently equation 5.1) was developed for application to munitions cooperatively attacking a target declared by another munition. In applying the equation to the simulation, no distinction was made between a munition attacking a target

declared by another munition and a munition attacking a target it had declared on its own. In truth however, there is a significant difference. For a munition attacking a target that it has declared on its own, $P_{sa}$ should be:

$$P_{sa} = P_K \cdot P_{RT|TR} \tag{5.1}$$

Applying this logic to the computer simulation, the munition that initially sees a target will have the greatest $P_{sa}$, since all other munitions will have to consider the probability that they will not recognize the target if they attempt the attack.

The calculation of the attack benefit (equations 3.2 and 3.3) included a term comparing the time remaining in the simulation to the estimated time required to reach the target (equation 3.6). The intent of including this term was to prevent munitions from attempting to attack targets they could not reach. However, the term also had the effect of decreasing the benefit of attacking targets as time progressed. In addition, it was realized that a similar time term was included in the calculation of the probability of a successful attack. The $P_{ss}$ used in calculating $P_{sa}$ is calculated at a time $t_r - t_{ETA}$. If the munition cannot reach a given target in the time remaining in the simulation, the $P_{ss}$ after the attack will be less than zero. With this realization, the time term was removed from the attack benefit calculation and logic was added to the simulation setting $P_{sa}$ equal to zero if the $P_{ss}$ after the attack was less than zero. With these changes, a munition still does not attack a target it cannot reach, and the duplication of the time parameter is eliminated.

**5.5.2 Response Surface Methodology.** Originally, the number of targets attacked and the hit formula were optimized using RSM. The data obtained with the optimum settings for these combined responses did not meet expectations. Specifically,

the number of targets attacked was consistently less for cooperative behavior than for non-cooperative behavior. In reviewing the results and the process behind them, it appeared that optimizing with respect to the hit formula often drove many of the munitions to attack a single, high value target. It was decided that the hit formula was not necessarily the best response to optimize, since it was concerned with the number of attacks versus the number of kills. New responses were chosen and the RSM re-accomplished for a limited number of scenarios. In the second RSM, the hit formula response was replaced with the number of false target attacks. Thus, the two responses to be optimized were the number of targets killed and the number of false target attacks, with the objective of maximizing and minimizing the respective responses.

The design space was also refined to reflect the expected optimum settings of the pertinent factors $\xi$ and $\beta$. Results from the original experiments consistently showed the optimum value of $\xi$ to be at or near the lowest setting. Thus, the design space for $\xi$ was allowed to vary from 0 to 0.5 (in natural variables). Since the number of targets killed (with no distinction between high priority and low priority targets) was used as a response in the new optimization, it was expected that the value of $\beta$ would be at the upper limit. Consequently, $\beta$ was allowed to range from 0.5 to 1.

**5.5.3 Results.** The results from the revised experiments and simulation were much more attractive than the original results. As shown in Figure 5.6, cooperative behavior provides mixed results compared to non-cooperative behavior. Cooperative engagement alone performs worse than all the other cooperation algorithms, including no cooperation at all. This can be explained in part by the simulation setup. In all scenarios, the munitions have a 50% overlap in coverage. Thus, even in the non-cooperating

scenarios, two munitions may see a target and attack it—providing a killed target despite the low lethality warheads.

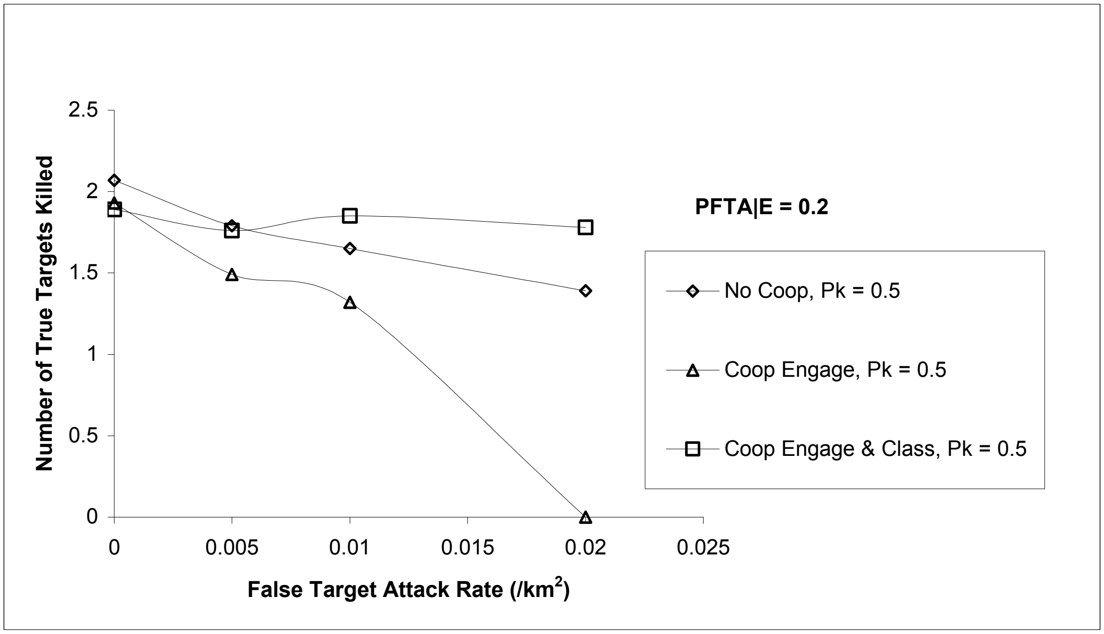



**Figure 5.6 Number of Targets Killed with Revised Experiments**

Another reason the cooperative engagement algorithm does not perform as well as the non-cooperating algorithm is, as with the original experiments, too many munitions attack the same targets (see Figure 5.7). In fact, this behavior has enormous consequences as the number of false targets increases. As $\alpha$ increases, the cooperative engagement algorithm calls more and more munitions to attack false targets. Eventually (at $\alpha$ greater than 0.01), the number of false target attacks begins to outweigh the number of true targets killed, and the RSM determines optimum settings that simply prohibit munitions from executing any attacks. The fact that similar behavior is not seen in the non-cooperating algorithm can be attributed to the higher number of killed targets that the non-cooperating algorithm produces. With the non-cooperating algorithm the number of killed targets still outweighs the number of false target attacks, though as $\alpha$ continues to

increase, behavior similar to that of the cooperative engagement algorithm would be expected.

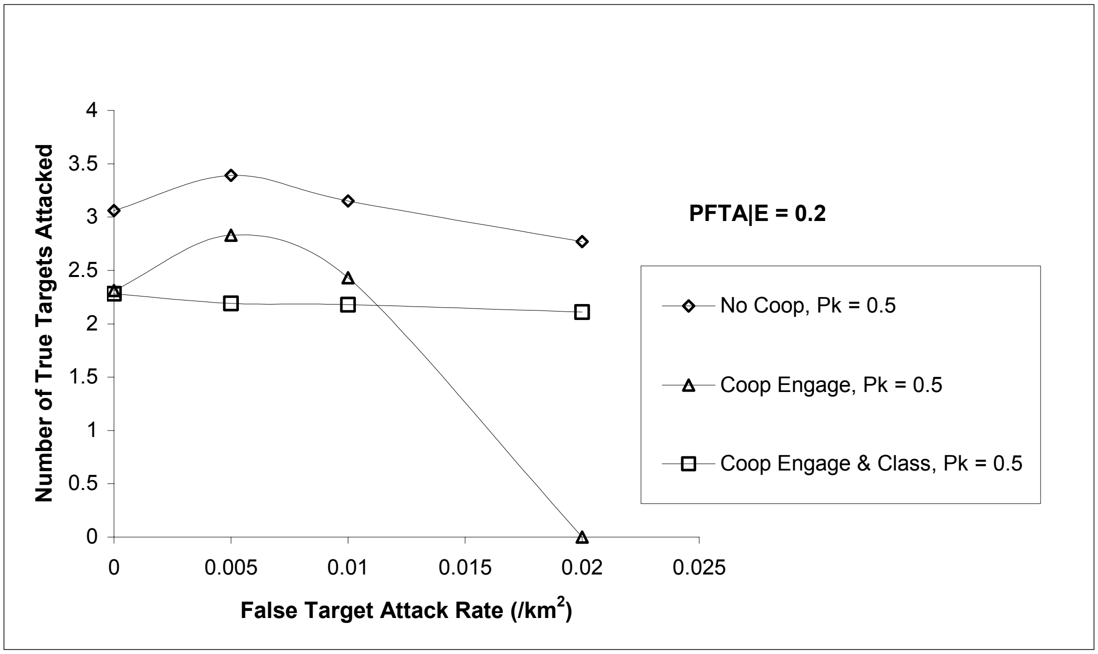



**Figure 5.7 Number of Targets Attacked with Revised Experiments**

The cooperative classification algorithm also kills fewer targets than the non-cooperative algorithm when the false target attack rate is less than 0.005 (see Figure 5.6). This is due to the limited number of attacks executed on valid targets. Since the cooperative classification algorithm requires a higher confidence level for classifying objects, the munitions spend more time looking at targets and making sure of their identification. This limits the amount of area they search and consequently limits the number of targets they encounter and attack. However, as $\alpha$ increases above 0.005, the cooperative classification algorithm begins to outperform the other algorithms. This is due to the effective $\alpha$. From Figure 5.8, we see that the number of false target attacks in both the non-cooperative and cooperative engagement algorithms steadily increases with the false target attack rate. Cooperative classification however, appears to significantly

reduce and stabilize the effective false target attack rate. Thus, more munitions are kept available to attack true targets and provide more kills than the other algorithms.

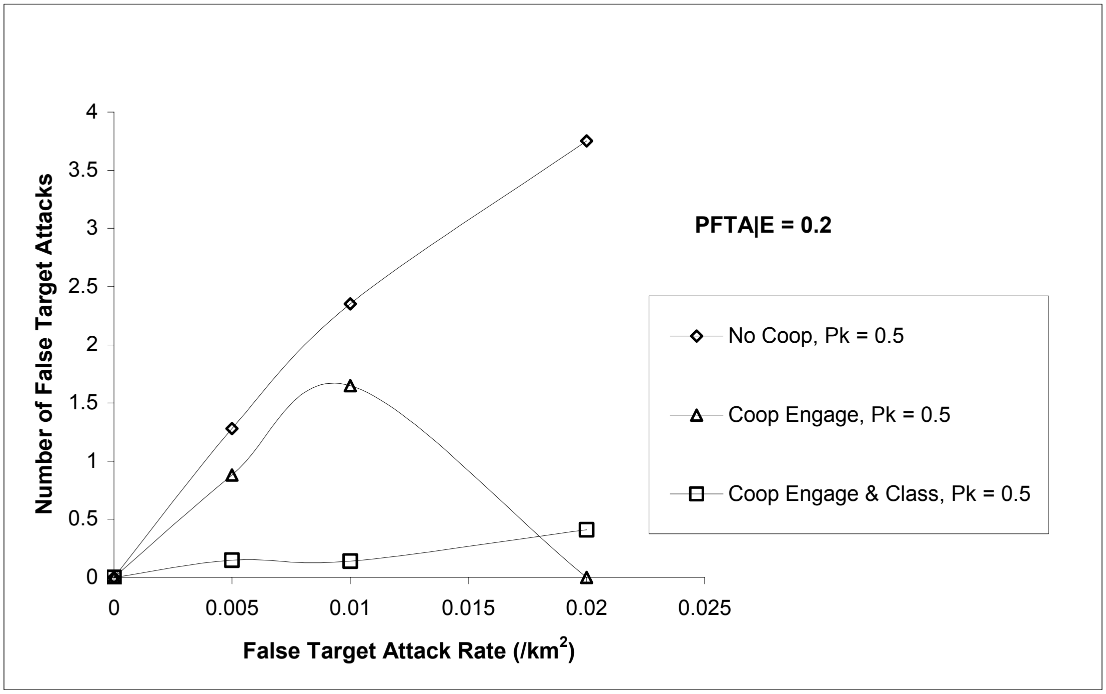



**Figure 5.8 Number of False Target Attacks with Revised Experiments**

# VI. Conclusions and Recommendations

This research investigated the use of cooperative behavior in autonomous wide area search munitions. Generic munitions and scenarios were used so that the results would have broad applicability. Two categories of conclusions may be drawn from these results: those pertaining to the application of cooperative behavior and those pertaining to the process used in this research.

It is generally believed that cooperative behavior will improve group performance. This research shows that conclusion is not necessarily true. The results depend on the form of cooperative behavior used and the specific scenario characteristics (namely the false target attack rate). Cooperative attack by itself does not improve performance over non-cooperating munitions. In fact, it does quite the opposite. Cooperative attack brings more munitions to bear against fewer targets. Thus, the system as a whole attacks and kills fewer targets. In addition, as the false target attack rate increases, the possibility that a misidentified non-target will draw away many of the munitions becomes very real. Eventually, the possibility of the munitions being wasted on non-targets becomes so great that it is better for the munitions to simply not attack anything at all.

Cooperative classification however, offers more promise. At low false target attack rates, cooperatively classifying munitions perform nearly as well (though still slightly worse) than non-cooperating munitions. The benefit to cooperative classification becomes apparent as the false target attack rate becomes large. While the other cooperation algorithms see decreasing performance as the false target attack rate increases, the performance with cooperative classification remains relatively constant.

This is due to the remarkable ability of cooperative classification to discern non-targets from real targets. Even at high false target attack rates, cooperative classification allows the munitions to identify objects with a very high degree of accuracy. Non-targets are not attacked nearly as often as they are with the other cooperation algorithms and thus more munitions are kept available to attack real targets.

The improved performance seen with cooperative classification alludes to excess capacity in the system as a whole. How much more is the system capable of when cooperative classification is employed? The precise boundaries of these expanded capabilities can be explored by varying different parameters such as the number and grouping of munitions and the search pattern and formation of those munitions. In particular, the overlap in the munitions' fields of regard could be removed. This would allow the munitions to search a larger area, or allow the same area to be searched with fewer munitions.

The consistency of performance exhibited with cooperative classification hints at a robustness across scenarios. The actual extent of this robustness should be investigated by varying the battlefield parameters. Parameters of particular interest would be the distribution of targets (i.e. uniform versus clustered), relative density of high versus low priority targets, etc.

Another area that may be of interest is the task allocation. Both Gillen's research (3) and this research used RSM to optimize the decision rule parameters off-line. However, this research also used an on-line optimization to actually allocate tasks as a group. Such a form of task allocation would most likely be more costly and prone to failure since it would require communicating more information between individual

munitions. An interesting question is whether the on-line optimization improves performance enough to warrant such risks or costs.

For simplicity, this research purposely ignored some limitations in the simulation which future research may want to address. Some of these limitations are listed below.

- Communications. Although bad information could be communicated in the current simulation, every munition received the same information. A more realistic scenario would limit the communications range of the munitions, allowing some munitions to not receive any information about some targets. This would lead to sub-optimal solutions to the transshipment problem.
- Moving Targets. The current simulation only employs stationary targets. Moving targets could significantly complicate a simulation because it introduces the problem of target registry and how munitions determine if sighted targets are ones that have already been sighted, but it is a problem that will eventually need to be addressed—most of our enemies do not stand still and wait for us to shoot them.
- Link classifications with viewing angle. In reality, the likelihood of a munition correctly identifying an object would depend on the viewing angle. In the simulation used in this research, the two were separate. A confidence level was constructed based on the viewing angle, but the classification of the object was based on a simple random draw that did not change with the viewing angle.

Finally, one important lesson was learned in this research regarding the process itself. While a wide variety of methods can be used to obtain equally valid results, the

actual applicability of those results depends greatly the method used. In this research, the original choice of responses used in the RSM was poor. The results obtained from the original runs provided some insight into the effects of cooperative behavior, but they did not provide as much information as was desired—they could not since the responses that were optimized did not address the same issues about which we wanted to draw conclusions. Thus, the importance of thorough planning was inadvertently illustrated.

# Appendix A:  Test Matrices

## Table A.1.  Test Matrix for Original Objective

Test Matrix for the Original Objective

| Coop Factor | |
|---|---|
| 0 | No Coop |
| 1 | Coop Engage |
| 2 | Coop Engage & Class |

$P_{FTA|E}$ 0.2

| $P_k$ | $\eta_T$ | $\eta_{FT}$ | $\alpha$ | Coop Factor | Significant Factors | | # Killed Targets | | | Formula | | | Scenario |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Tgt2Value | SearchWeight | Average | Low 95% | Up 95% | Average | Low 95% | Up 95% | |
| 0.5 | 0.00774 | 0 | 0 | 0 | -1 | -1 | 1.29 | 1.1539 | 1.4261 | 6.28 | 5.92 | 6.64 | 41 |
| | | | | 1 | | -1 | 1.09 | 0.9804 | 1.1996 | 12.29 | 11.315 | 13.265 | 42 |
| | | | | 2 | -1 | -1 | 1.14 | 1.017 | 1.263 | 11.96 | 10.846 | 13.074 | 43 |
| | | 0.025 | 0.005 | 0 | | -1 | 1.19 | 1.047 | 1.333 | 5.07 | 4.567 | 5.573 | 53 |
| | | | | 1 | | -1 | 1.08 | 0.9487 | 1.2113 | 10.11 | 8.837 | 11.383 | 54 |
| | | | | 2 | | -1 | 1.04 | 0.924 | 1.156 | 11.68 | 10.509 | 12.851 | 55 |
| | | 0.05 | 0.01 | 0 | | -1 | 1.12 | 0.9839 | 1.2561 | 3.98 | 3.368 | 4.592 | 59 |
| | | | | 1 | -1 | -1 | 1.04 | 0.9311 | 1.1489 | 9.42 | 7.987 | 10.853 | 60 |
| | | | | 2 | | -1 | 1.11 | 1.0049 | 1.2151 | 12.65 | 11.579 | 13.721 | 61 |
| | | 0.1 | 0.02 | 0 | -1 | -1 | 0.98 | 0.8363 | 1.1237 | 0.98 | 0.279 | 1.681 | 47 |
| | | | | 1 | 1 | -1 | 1.17 | 1.0288 | 1.3112 | 4.12 | 2.908 | 5.332 | 48 |
| | | | | 2 | | -1 | 1.09 | 0.9769 | 1.2031 | 11.51 | 10.319 | 12.701 | 49 |

| $P_k$ | $\eta_T$ | $\eta_{FT}$ | $\alpha$ | Coop Factor | Significant Factors | | # Killed Targets | | | Formula | | | Scenario |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Tgt2Value | SearchWeight | Average | Low 95% | Up 95% | Average | Low 95% | Up 95% | |
| 0.8 | 0.00774 | 0 | 0 | 0 | | -1 | 1.63 | 1.5115 | 1.7485 | 6.13 | 5.756 | 6.504 | 44 |
| | | | | 1 | -1 | -1 | 1.34 | 1.2375 | 1.4425 | 11.99 | 11.327 | 12.653 | 45 |
| | | | | 2 | -1 | -1 | 1.24 | 1.1303 | 1.3497 | 11.2 | 10.377 | 12.023 | 46 |
| | | 0.025 | 0.005 | 0 | 1 | -1 | 2.53 | 2.3462 | 2.7138 | 5.42 | 4.7982 | 6.042 | 56 |
| | | | | 1 | | -1 | 1.32 | 1.2076 | 1.4324 | 10.08 | 9.1456 | 11.014 | 57 |
| | | | | 2 | | -1 | 1.23 | 1.125 | 1.335 | 10.54 | 9.6923 | 11.388 | 58 |
| | | 0.05 | 0.01 | 0 | 1 | -1 | 2.16 | 1.9778 | 2.3422 | 2.9 | 2.2824 | 3.518 | 62 |
| | | | | 1 | 1 | -1 | 1.56 | 1.4238 | 1.6962 | 6.77 | 5.6045 | 7.935 | 63 |
| | | | | 2 | 1 | -1 | 1.83 | 1.7006 | 1.9594 | 9.77 | 9.0029 | 10.537 | 64 |
| | | 0.1 | 0.02 | 0 | 1 | -1 | 1.69 | 1.5036 | 1.8764 | 0.2 | -0.607 | 1.0066 | 50 |
| | | | | 1 | | -1 | 1 | 0.8837 | 1.1163 | 4.77 | 3.407 | 6.1328 | 51 |
| | | | | 2 | 1 | -1 | 1.69 | 1.5499 | 1.8301 | 8.64 | 7.606 | 9.6737 | 52 |

## Table A.2. Test Matrix for Revised Objective

Test Matrix for Revised Objective

| Coop Factor | |
|---|---|
| 0 | No Coop |
| 1 | Coop Engage |
| 2 | Coop Engage & Class |

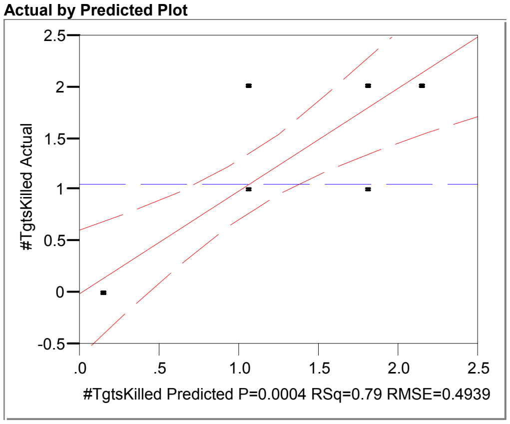
$P_{FTA|E}$ 0.2

| $P_k$ | $\eta_T$ | $\eta_{FT}$ | $\alpha$ | Coop Factor | Significant Factors | | # Killed Targets | | # False Tgt Attacks | | Scenario |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Tgt2Value | SearchWeight | Average | $\sigma$ | Average | $\sigma$ | |
| 0.5 | 0.00774 | 0 | 0 | 0 | | -0.5769 | 2.07 | 0.819645 | 0 | 0 | 80 |
| | | | | 1 | | -0.5 | 1.93 | 0.768772 | 0 | 0 | 81 |
| | | | | 2 | | -0.2692 | 1.89 | 0.851558 | 0 | 0 | 82 |
| | | 0.025 | 0.005 | 0 | | 0.60331 | 1.79 | 0.97747 | 1.28 | 0.82975 | 86 |
| | | | | 1 | | 0.51187 | 1.49 | 1.01 | 0.88 | 0.890806 | 87 |
| | | | | 2 | | -0.4333 | 1.76 | 0.78005 | 0.15 | 0.609272 | 88 |
| | | 0.05 | 0.01 | 0 | | 0.72123 | 1.65 | 1.04809 | 2.35 | 1.17529 | 92 |
| | | | | 1 | | 0.66918 | 1.32 | 0.94152 | 1.65 | 1.05768 | 93 |
| | | | | 2 | | -0.4091 | 1.85 | 0.77035 | 0.14 | 0.40252 | 94 |
| | | 0.1 | 0.02 | 0 | | 0.88915 | 1.39 | 1.00398 | 3.72 | 1.31871 | 95 |
| | | | | 1 | | 1 | 0 | 0 | 0 | 0 | 96 |
| | | | | 2 | 1 | -0.42 | 1.78 | 0.73278 | 0.41 | 0.93306 | 97 |

# Appendix B: Sample Designs

## B.1 Sample Design – $\beta$ is insignificant:

**Least Squares Fit**
**Response #TgtsKilled**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.792526 |
| RSquare Adj | 0.735942 |
| Root Mean Square Error | 0.493902 |
| Mean of Response | 1.066667 |
| Observations (or Sum Wgts) | 15 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 3 | 10.250000 | 3.41667 | 14.0062 |
| Error | 11 | 2.683333 | 0.24394 | Prob > F |
| C. Total | 14 | 12.933333 | | 0.0004 |

**Lack Of Fit**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Lack Of Fit | 1 | 1.3500000 | 1.35000 | 10.1250 |
| Pure Error | 10 | 1.3333333 | 0.13333 | Prob > F |
| Total Error | 11 | 2.6833333 | | 0.0098 |
| | | | | Max RSq |
| | | | | 0.8969 |

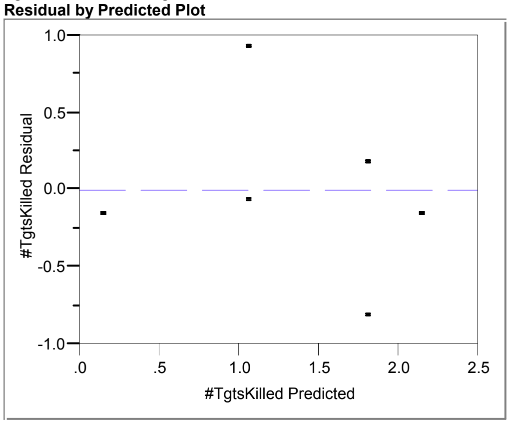
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | 1.0666667 | 0.127525 | 8.36 | <.0001 |
| Tgt2Value | -0.083333 | 0.142577 | -0.58 | 0.5707 |
| SearchWeight | -0.916667 | 0.142577 | -6.43 | <.0001 |
| Tgt2Value*SearchWeight | 0.0833333 | 0.142577 | 0.58 | 0.5707 |

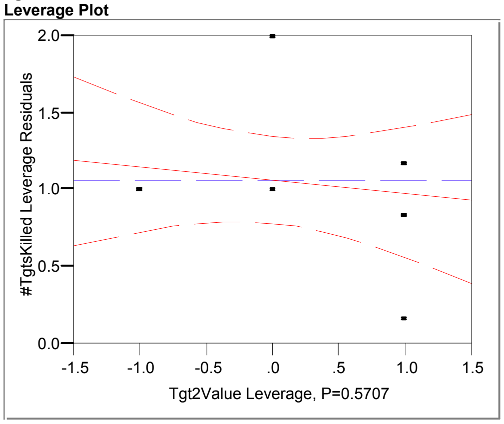
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value | 1 | 1 | 0.083333 | 0.3416 | 0.5707 |
| SearchWeight | 1 | 1 | 10.083333 | 41.3354 | <.0001 |
| Tgt2Value*SearchWeight | 1 | 1 | 0.083333 | 0.3416 | 0.5707 |

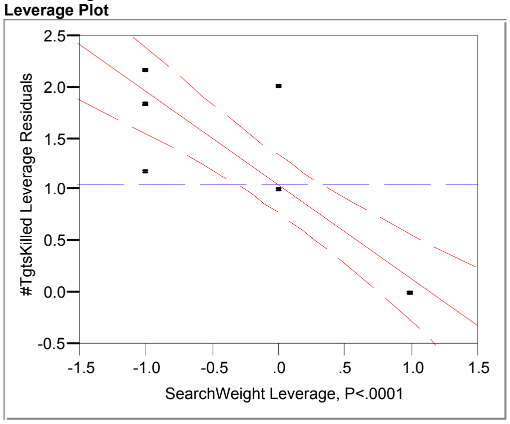
**Residual by Predicted Plot**



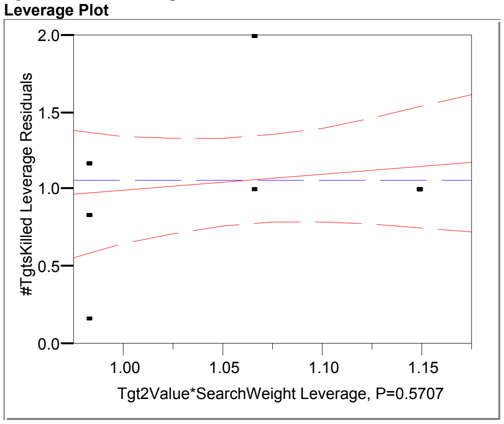
**Tgt2Value**
**Leverage Plot**



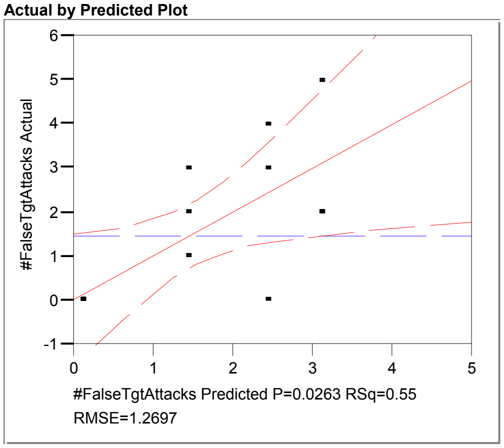

**SearchWeight**
**Leverage Plot**




**Tgt2Value*SearchWeight**
**Leverage Plot**



**Response #FalseTgtAttacks**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.553691 |
| RSquare Adj | 0.431971 |
| Root Mean Square Error | 1.269693 |
| Mean of Response | 1.466667 |
| Observations (or Sum Wgts) | 15 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 3 | 22.000000 | 7.33333 | 4.5489 |
| Error | 11 | 17.733333 | 1.61212 | Prob > F |
| C. Total | 14 | 39.733333 | | 0.0263 |

**Lack Of Fit**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Lack Of Fit | 1 | 1.066667 | 1.06667 | 0.6400 |
| Pure Error | 10 | 16.666667 | 1.66667 | Prob > F |
| Total Error | 11 | 17.733333 | | 0.4423 |
| | | | | Max RSq |
| | | | | 0.5805 |

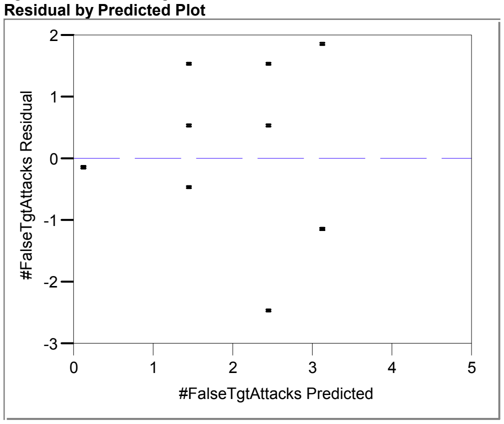
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | 1.4666667 | 0.327833 | 4.47 | 0.0009 |
| Tgt2Value | -0.166667 | 0.366529 | -0.45 | 0.6582 |
| SearchWeight | -1.333333 | 0.366529 | -3.64 | 0.0039 |
| Tgt2Value*SearchWeight | 0.1666667 | 0.366529 | 0.45 | 0.6582 |

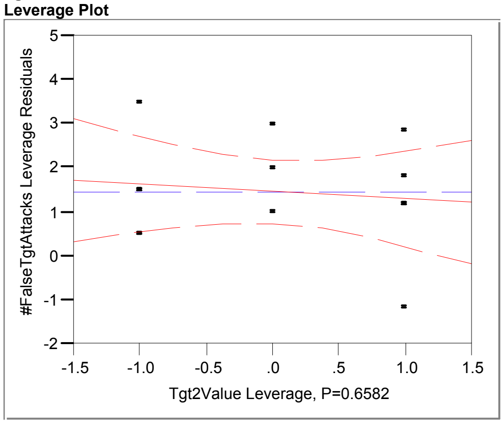
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value | 1 | 1 | 0.333333 | 0.2068 | 0.6582 |
| SearchWeight | 1 | 1 | 21.333333 | 13.2331 | 0.0039 |
| Tgt2Value*SearchWeight | 1 | 1 | 0.333333 | 0.2068 | 0.6582 |

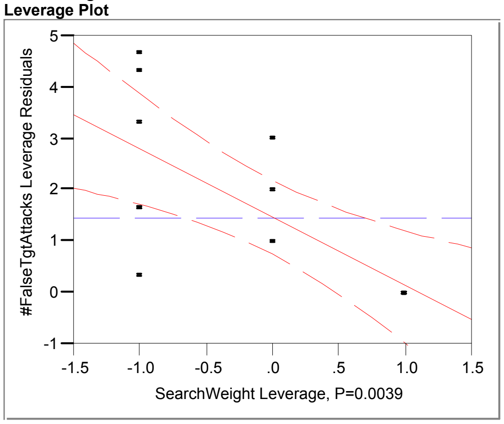
**Residual by Predicted Plot**



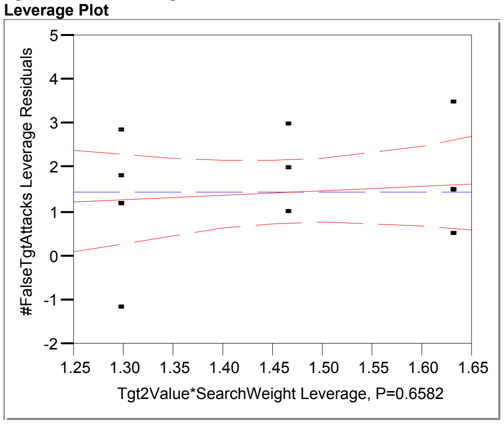
**Tgt2Value**
**Leverage Plot**




**SearchWeight**
**Leverage Plot**

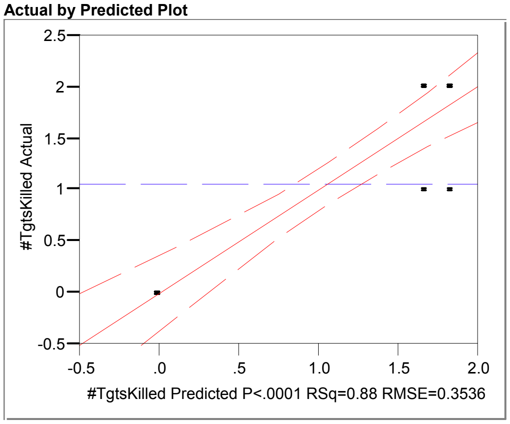



**Tgt2Value*SearchWeight**
**Leverage Plot**



## B.2 Sample Final Model – $\beta$ is insignificant:

**Least Squares Fit**
**Response #TgtsKilled**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.884021 |
| RSquare Adj | 0.864691 |
| Root Mean Square Error | 0.353553 |
| Mean of Response | 1.066667 |
| Observations (or Sum Wgts) | 15 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 2 | 11.433333 | 5.71667 | 45.7333 |
| Error | 12 | 1.500000 | 0.12500 | Prob > F |
| C. Total | 14 | 12.933333 | | <.0001 |

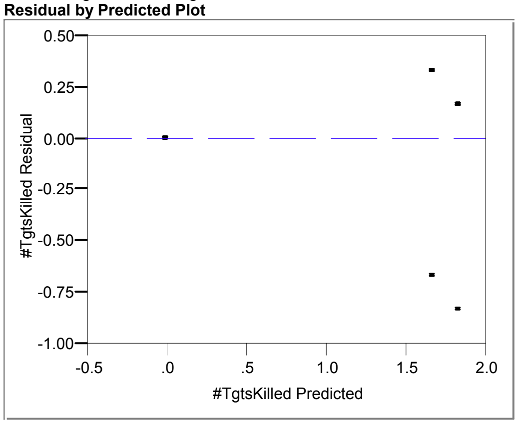
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | 1.6666667 | 0.204124 | 8.16 | <.0001 |
| SearchWeight | -0.916667 | 0.102062 | -8.98 | <.0001 |
| SearchWeight*SearchWeight | -0.75 | 0.228218 | -3.29 | 0.0065 |

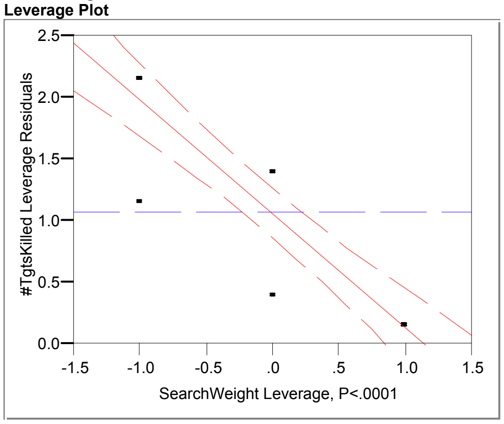
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| SearchWeight | 1 | 1 | 10.083333 | 80.6667 | <.0001 |
| SearchWeight*SearchWeight | 1 | 1 | 1.350000 | 10.8000 | 0.0065 |

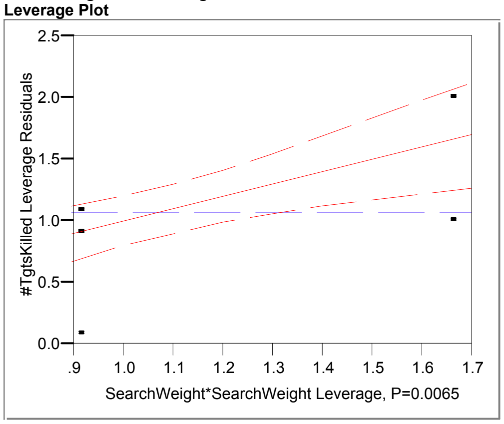
**Residual by Predicted Plot**



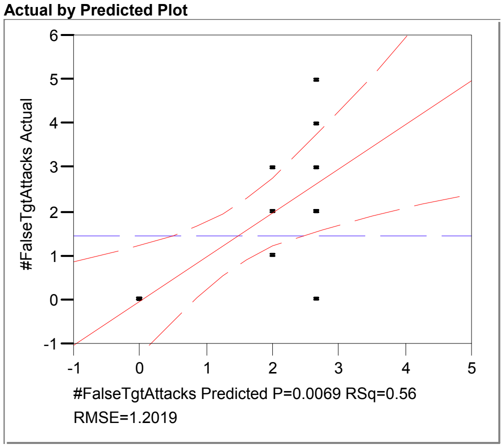
**SearchWeight**
**Leverage Plot**




**SearchWeight*SearchWeight**
**Leverage Plot**




**Response #FalseTgtAttacks**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.563758 |
| RSquare Adj | 0.491051 |
| Root Mean Square Error | 1.20185 |
| Mean of Response | 1.466667 |
| Observations (or Sum Wgts) | 15 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 2 | 22.400000 | 11.2000 | 7.7538 |
| Error | 12 | 17.333333 | 1.4444 | Prob > F |
| C. Total | 14 | 39.733333 | | 0.0069 |

**Parameter Estimates**

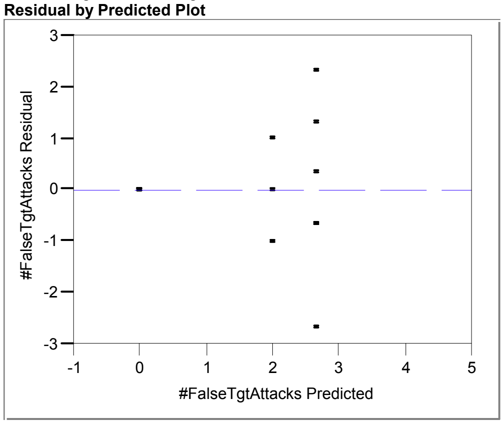
| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | 2 | 0.693889 | 2.88 | 0.0138 |
| SearchWeight | -1.333333 | 0.346944 | -3.84 | 0.0023 |
| SearchWeight*SearchWeight | -0.666667 | 0.775791 | -0.86 | 0.4070 |

**Effect Tests**

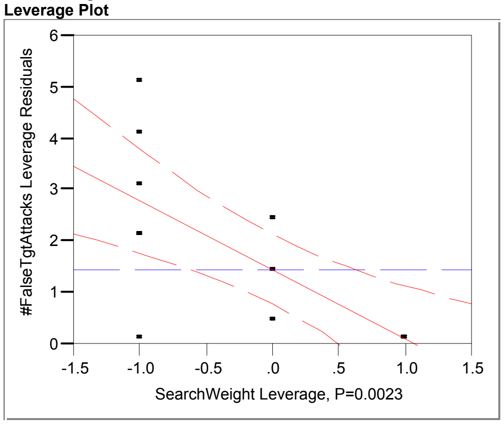
| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| SearchWeight | 1 | 1 | 21.333333 | 14.7692 | 0.0023 |
| SearchWeight*SearchWeight | 1 | 1 | 1.066667 | 0.7385 | 0.4070 |

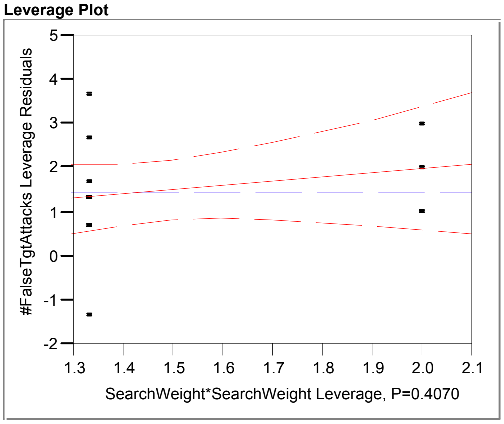
**Residual by Predicted Plot**

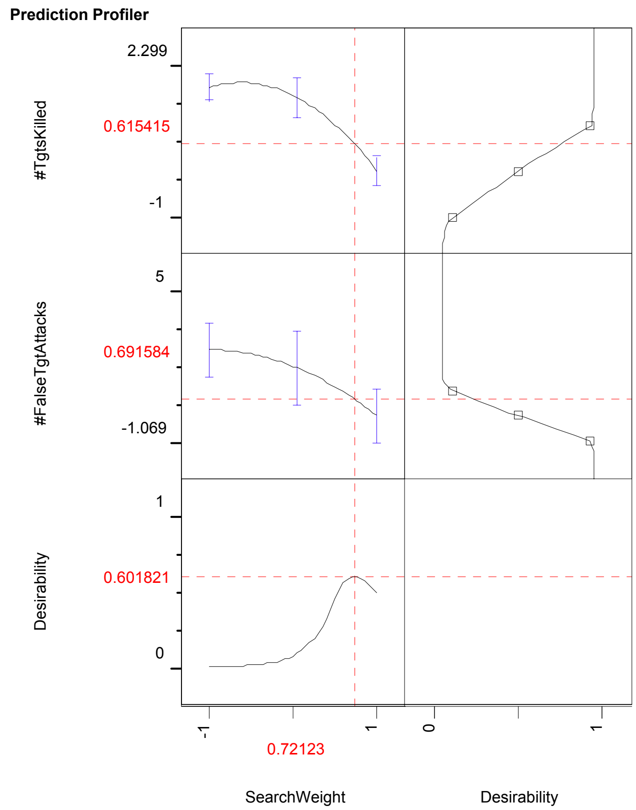



**SearchWeight**
**Leverage Plot**



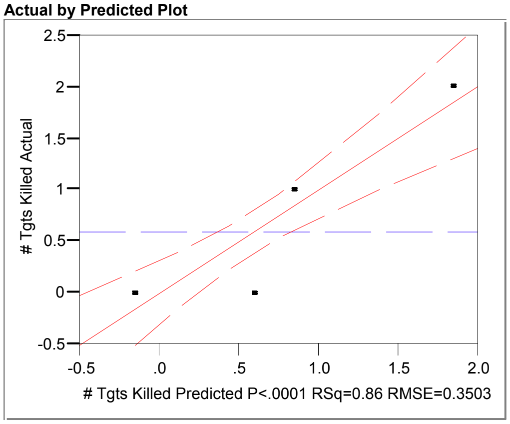

**SearchWeight*SearchWeight**
**Leverage Plot**




Prediction Profiler
2.299
0.615415
-1
#TgtsKilled
5
0.691584
-1.069
#FalseTgtAttacks
1
0.601821
0
Desirability
-1
1
0.72123
0
1
SearchWeight
Desirability

## B.3 Sample Screening Design: $\xi$ and $\beta$ are significant:

**Least Squares Fit**
**Response # Tgts Killed**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.859375 |
| RSquare Adj | 0.821023 |
| Root Mean Square Error | 0.350325 |
| Mean of Response | 0.6 |
| Observations (or Sum Wgts) | 15 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 3 | 8.2500000 | 2.75000 | 22.4074 |
| Error | 11 | 1.3500000 | 0.12273 | Prob > F |
| C. Total | 14 | 9.6000000 | | <.0001 |

**Lack Of Fit**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Lack Of Fit | 1 | 1.3500000 | 1.35000 | . |
| Pure Error | 10 | 0.0000000 | 0.00000 | Prob > F |
| Total Error | 11 | 1.3500000 | | . |
| | | | | Max RSq |
| | | | | 1.0000 |

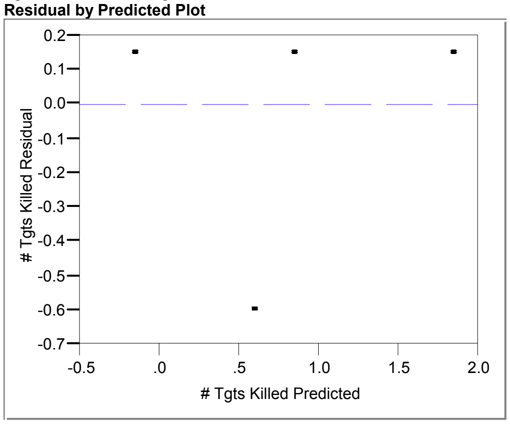
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | 0.6 | 0.090453 | 6.63 | <.0001 |
| Tgt2Value | 0.25 | 0.10113 | 2.47 | 0.0310 |
| SearchWeight | -0.75 | 0.10113 | -7.42 | <.0001 |
| Tgt2Value*SearchWeight | -0.25 | 0.10113 | -2.47 | 0.0310 |

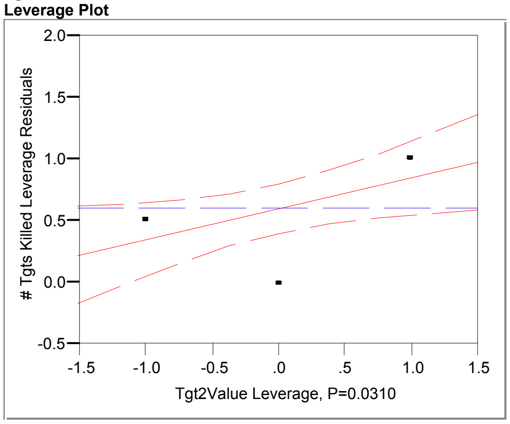
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value | 1 | 1 | 0.7500000 | 6.1111 | 0.0310 |
| SearchWeight | 1 | 1 | 6.7500000 | 55.0000 | <.0001 |
| Tgt2Value*SearchWeight | 1 | 1 | 0.7500000 | 6.1111 | 0.0310 |

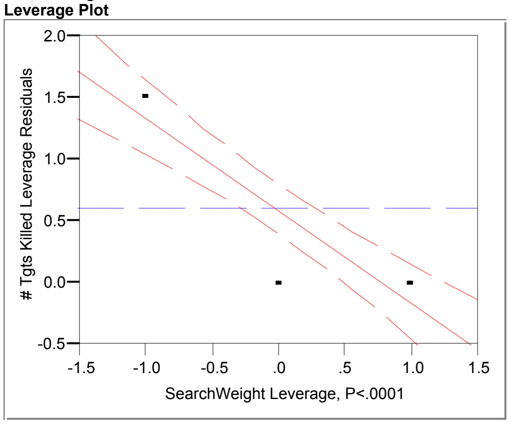
**Residual by Predicted Plot**



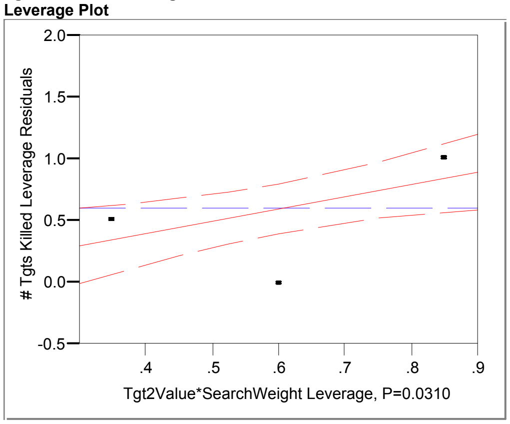
**Tgt2Value**
**Leverage Plot**



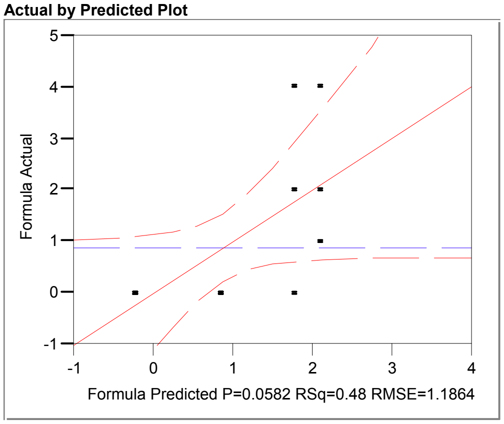

**SearchWeight**
**Leverage Plot**




**Tgt2Value*SearchWeight**
**Leverage Plot**



**Response Formula**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.47926 |
| RSquare Adj | 0.33724 |
| Root Mean Square Error | 1.186413 |
| Mean of Response | 0.866667 |
| Observations (or Sum Wgts) | 15 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 3 | 14.250000 | 4.75000 | 3.3746 |
| Error | 11 | 15.483333 | 1.40758 | Prob > F |
| C. Total | 14 | 29.733333 | | 0.0582 |

**Lack Of Fit**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Lack Of Fit | 1 | 2.816667 | 2.81667 | 2.2237 |
| Pure Error | 10 | 12.666667 | 1.26667 | Prob > F |
| Total Error | 11 | 15.483333 | | 0.1668 |
| | | | | Max RSq |
| | | | | 0.5740 |

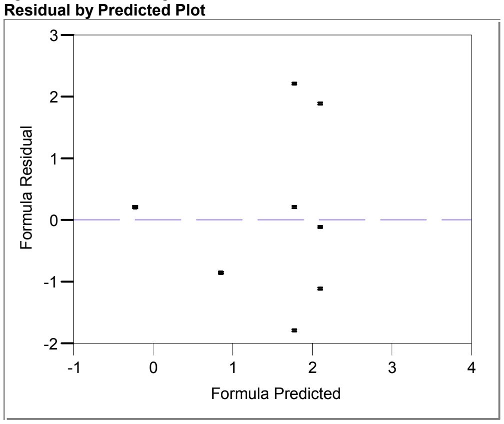
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | 0.8666667 | 0.306331 | 2.83 | 0.0164 |
| Tgt2Value | -0.083333 | 0.342488 | -0.24 | 0.8122 |
| SearchWeight | -1.083333 | 0.342488 | -3.16 | 0.0090 |
| Tgt2Value*SearchWeight | 0.0833333 | 0.342488 | 0.24 | 0.8122 |

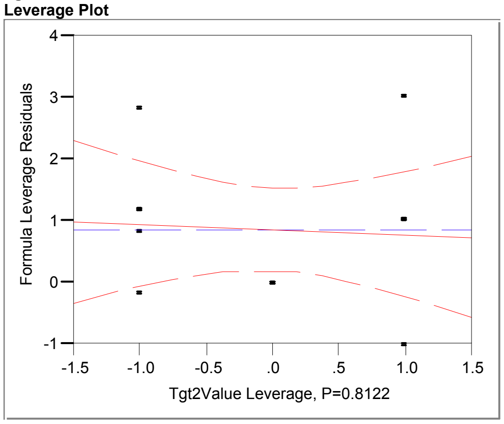
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value | 1 | 1 | 0.083333 | 0.0592 | 0.8122 |
| SearchWeight | 1 | 1 | 14.083333 | 10.0054 | 0.0090 |
| Tgt2Value*SearchWeight | 1 | 1 | 0.083333 | 0.0592 | 0.8122 |

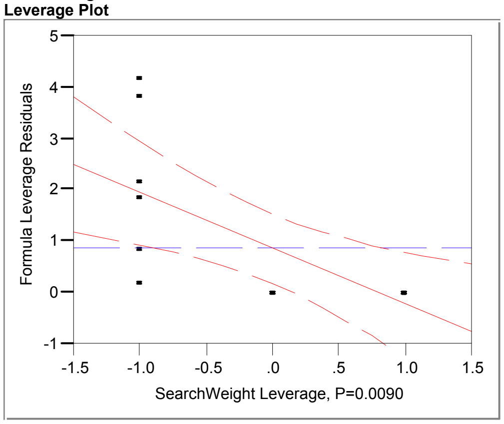
**Residual by Predicted Plot**



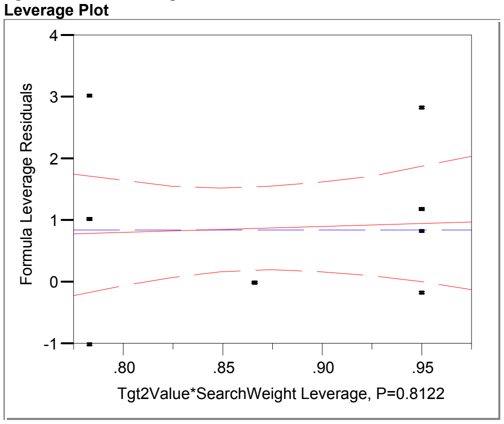
**Tgt2Value**
**Leverage Plot**




**SearchWeight**
**Leverage Plot**

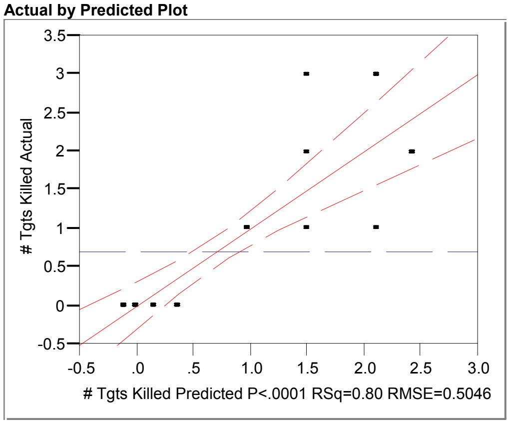



**Tgt2Value*SearchWeight**
**Leverage Plot**



## B.4 Sample Central Composite Design: $\xi$ and $\beta$ are significant:

**Least Squares Fit**
**Response # Tgts Killed**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.798609 |
| RSquare Adj | 0.75988 |
| Root Mean Square Error | 0.504622 |
| Mean of Response | 0.6875 |
| Observations (or Sum Wgts) | 32 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 5 | 26.254275 | 5.25085 | 20.6204 |
| Error | 26 | 6.620725 | 0.25464 | Prob > F |
| C. Total | 31 | 32.875000 | | <.0001 |

**Lack Of Fit**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Lack Of Fit | 3 | 1.9540585 | 0.651353 | 3.2102 |
| Pure Error | 23 | 4.6666667 | 0.202899 | Prob > F |
| Total Error | 26 | 6.6207251 | | 0.0418 |
| | | | | Max RSq |
| | | | | 0.8580 |

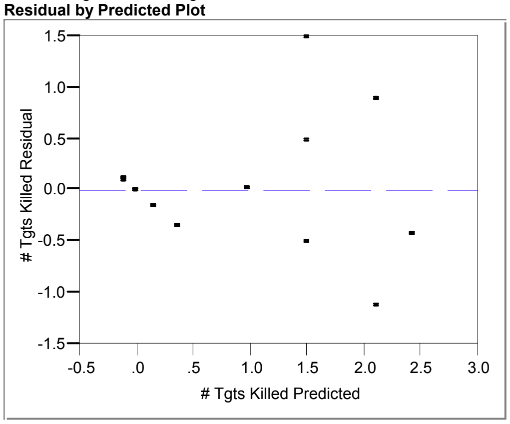
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | -3.38e-14 | 0.178411 | -0.00 | 1.0000 |
| Tgt2Value&RS | 0.4785534 | 0.103006 | 4.65 | <.0001 |
| SearchWeight&RS | -0.787479 | 0.103006 | -7.65 | <.0001 |
| Tgt2Value*SearchWeight | -0.25 | 0.145672 | -1.72 | 0.0980 |
| Tgt2Value*Tgt2Value | 0.4166667 | 0.126155 | 3.30 | 0.0028 |
| SearchWeight*SearchWeight | 0.5 | 0.126155 | 3.96 | 0.0005 |

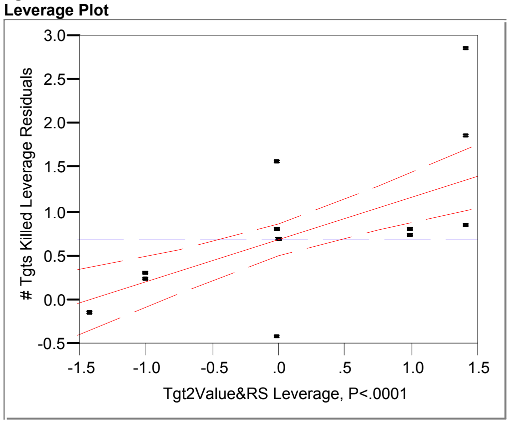
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value&RS | 1 | 1 | 5.496320 | 21.5844 | <.0001 |
| SearchWeight&RS | 1 | 1 | 14.882955 | 58.4463 | <.0001 |
| Tgt2Value*SearchWeight | 1 | 1 | 0.750000 | 2.9453 | 0.0980 |
| Tgt2Value*Tgt2Value | 1 | 1 | 2.777778 | 10.9085 | 0.0028 |
| SearchWeight*SearchWeight | 1 | 1 | 4.000000 | 15.7082 | 0.0005 |

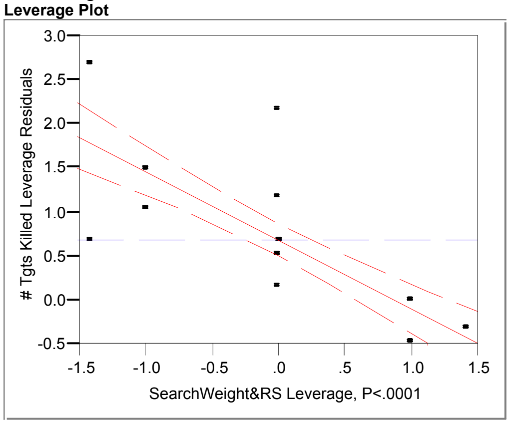
**Residual by Predicted Plot**



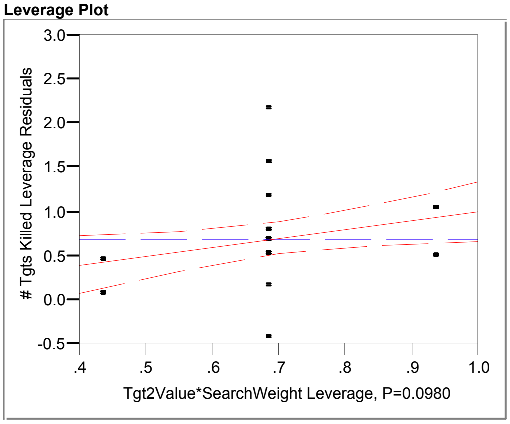
**Tgt2Value&RS**
**Leverage Plot**



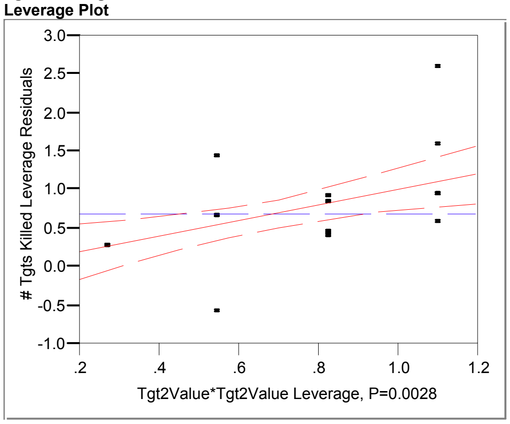

**SearchWeight&RS**
**Leverage Plot**



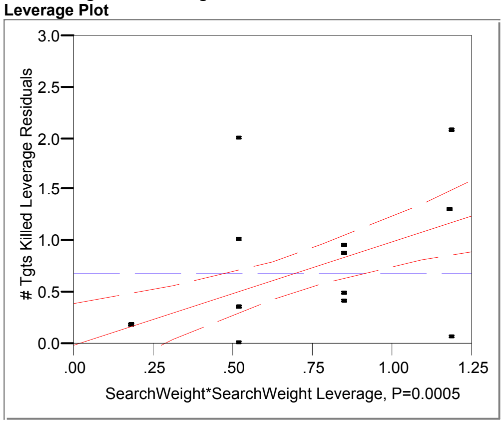

**Tgt2Value*SearchWeight**
**Leverage Plot**



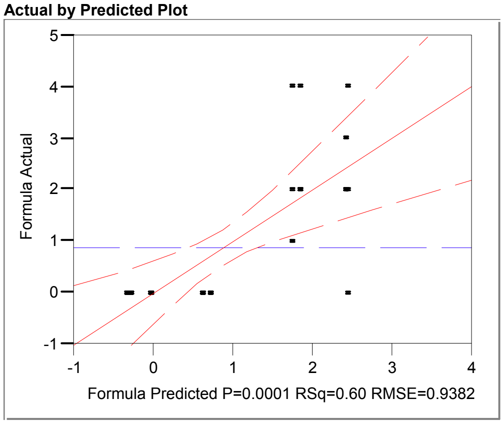
**Tgt2Value*Tgt2Value**
**Leverage Plot**




**SearchWeight*SearchWeight**
**Leverage Plot**




**Response Formula**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.602 |
| RSquare Adj | 0.525461 |
| Root Mean Square Error | 0.938186 |
| Mean of Response | 0.875 |
| Observations (or Sum Wgts) | 32 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 5 | 34.614977 | 6.92300 | 7.8653 |
| Error | 26 | 22.885023 | 0.88019 | Prob > F |
| C. Total | 31 | 57.500000 | | 0.0001 |

**Lack Of Fit**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Lack Of Fit | 3 | 6.885023 | 2.29501 | 3.2991 |
| Pure Error | 23 | 16.000000 | 0.69565 | Prob > F |
| Total Error | 26 | 22.885023 | | 0.0384 |
| | | | | Max RSq |
| | | | | 0.7217 |

**Parameter Estimates**

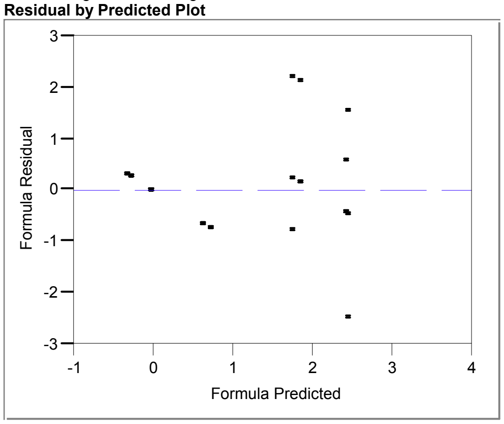
| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | -7.46e-14 | 0.331699 | -0.00 | 1.0000 |
| Tgt2Value&RS | 0.4297379 | 0.191506 | 2.24 | 0.0336 |
| SearchWeight&RS | -0.954146 | 0.191506 | -4.98 | <.0001 |
| Tgt2Value*SearchWeight | 0.0833333 | 0.270831 | 0.31 | 0.7608 |
| Tgt2Value*Tgt2Value | 0.625 | 0.234547 | 2.66 | 0.0131 |
| SearchWeight*SearchWeight | 0.5416667 | 0.234547 | 2.31 | 0.0291 |

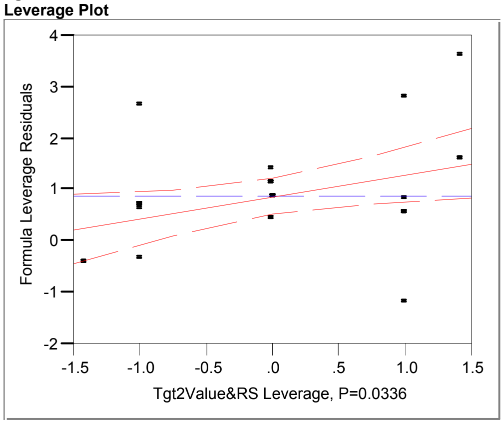
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value&RS | 1 | 1 | 4.432191 | 5.0355 | 0.0336 |
| SearchWeight&RS | 1 | 1 | 21.849453 | 24.8235 | <.0001 |
| Tgt2Value*SearchWeight | 1 | 1 | 0.083333 | 0.0947 | 0.7608 |
| Tgt2Value*Tgt2Value | 1 | 1 | 6.250000 | 7.1007 | 0.0131 |
| SearchWeight*SearchWeight | 1 | 1 | 4.694444 | 5.3334 | 0.0291 |

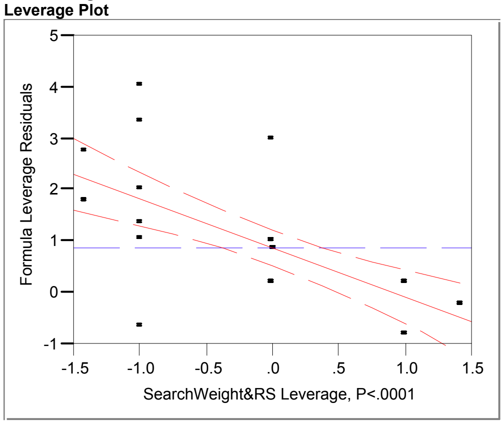
**Residual by Predicted Plot**



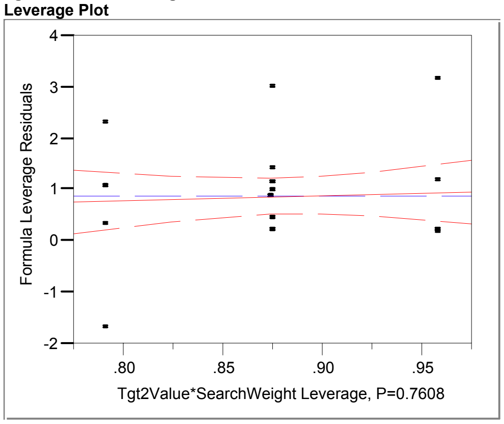

**Tgt2Value&RS**
**Leverage Plot**



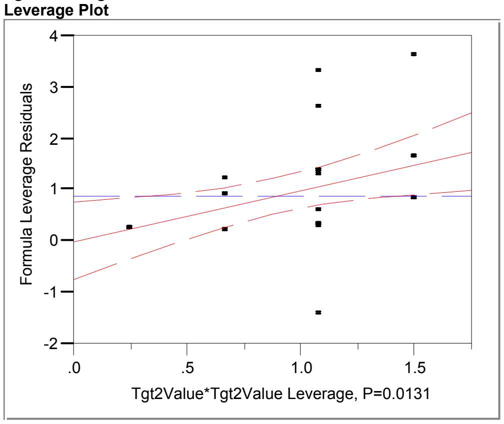

**SearchWeight&RS**
**Leverage Plot**



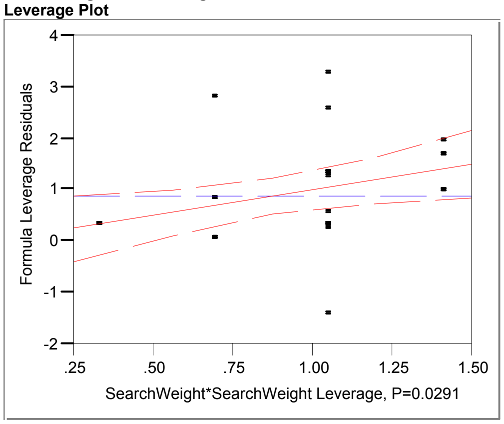
**Tgt2Value*SearchWeight**
**Leverage Plot**




**Tgt2Value*Tgt2Value**
**Leverage Plot**

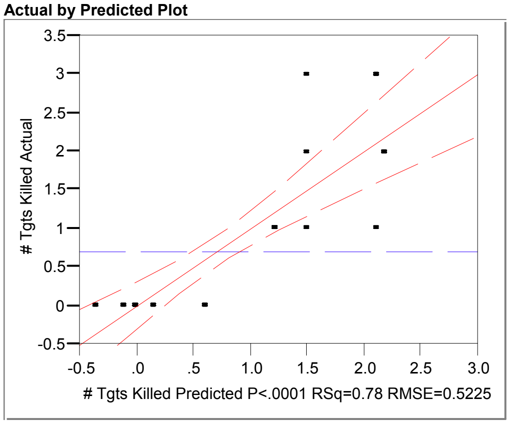



**SearchWeight*SearchWeight**
**Leverage Plot**



## B.5 Sample Final Design: $\xi$ and $\beta$ are significant:

**Least Squares Fit**
**Response # Tgts Killed**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.775795 |
| RSquare Adj | 0.74258 |
| Root Mean Square Error | 0.522484 |
| Mean of Response | 0.6875 |
| Observations (or Sum Wgts) | 32 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 4 | 25.504275 | 6.37607 | 23.3564 |
| Error | 27 | 7.370725 | 0.27299 | Prob > F |
| C. Total | 31 | 32.875000 | | <.0001 |

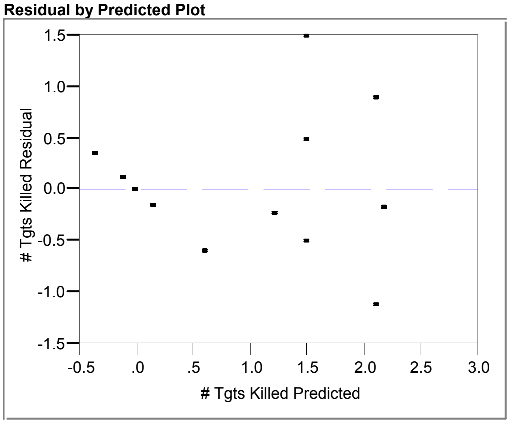
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | -4.8e-14 | 0.184726 | -0.00 | 1.0000 |
| Tgt2Value&RS | 0.4785534 | 0.106652 | 4.49 | 0.0001 |
| SearchWeight&RS | -0.787479 | 0.106652 | -7.38 | <.0001 |
| Tgt2Value*Tgt2Value | 0.4166667 | 0.130621 | 3.19 | 0.0036 |
| SearchWeight*SearchWeight | 0.5 | 0.130621 | 3.83 | 0.0007 |

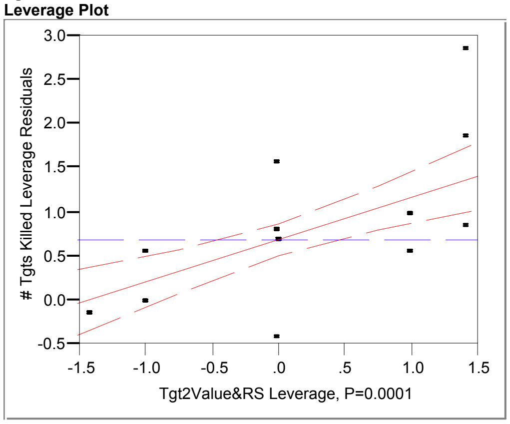
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value&RS | 1 | 1 | 5.496320 | 20.1338 | 0.0001 |
| SearchWeight&RS | 1 | 1 | 14.882955 | 54.5184 | <.0001 |
| Tgt2Value*Tgt2Value | 1 | 1 | 2.777778 | 10.1754 | 0.0036 |
| SearchWeight*SearchWeight | 1 | 1 | 4.000000 | 14.6526 | 0.0007 |

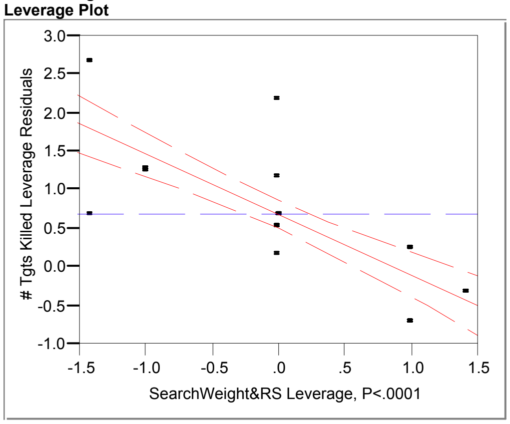
**Residual by Predicted Plot**



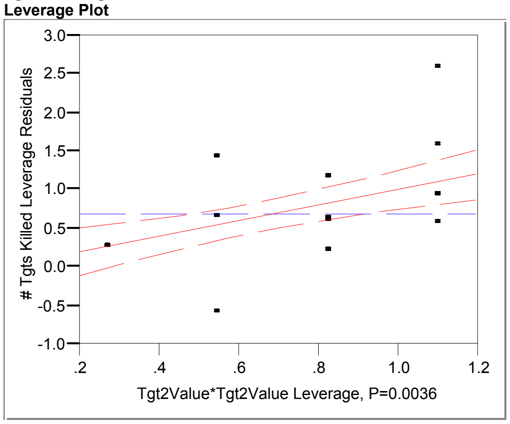
**Tgt2Value&RS**
**Leverage Plot**



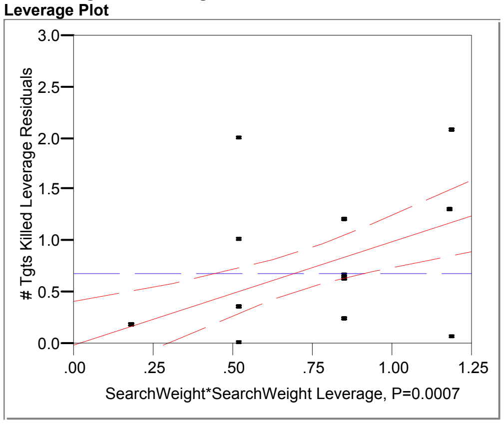

**SearchWeight&RS**
**Leverage Plot**



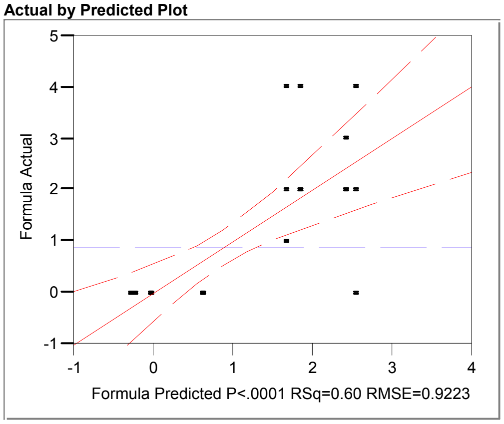

**Tgt2Value*Tgt2Value**
**Leverage Plot**



**SearchWeight*SearchWeight**
**Leverage Plot**




**Response Formula**
**Whole Model**
**Actual by Predicted Plot**




**Summary of Fit**

| | |
|---|---|
| RSquare | 0.60055 |
| RSquare Adj | 0.541373 |
| Root Mean Square Error | 0.922323 |
| Mean of Response | 0.875 |
| Observations (or Sum Wgts) | 32 |

**Analysis of Variance**

| Source | DF | Sum of Squares | Mean Square | F Ratio |
|---|---|---|---|---|
| Model | 4 | 34.531644 | 8.63291 | 10.1482 |
| Error | 27 | 22.968356 | 0.85068 | Prob > F |
| C. Total | 31 | 57.500000 | | <.0001 |

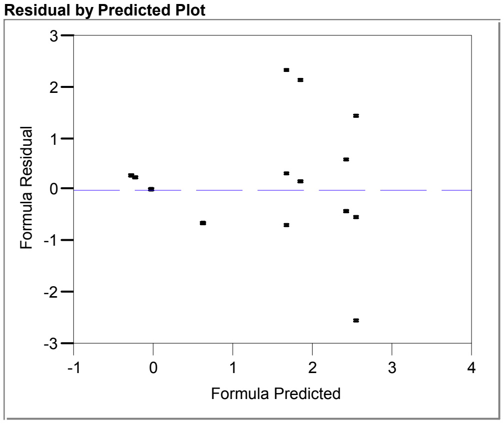
**Parameter Estimates**

| Term | Estimate | Std Error | t Ratio | Prob>\|t\| |
|---|---|---|---|---|
| Intercept | -6.93e-14 | 0.32609 | -0.00 | 1.0000 |
| Tgt2Value&RS | 0.4297379 | 0.188268 | 2.28 | 0.0306 |
| SearchWeight&RS | -0.954146 | 0.188268 | -5.07 | <.0001 |
| Tgt2Value*Tgt2Value | 0.625 | 0.230581 | 2.71 | 0.0115 |
| SearchWeight*SearchWeight | 0.5416667 | 0.230581 | 2.35 | 0.0264 |

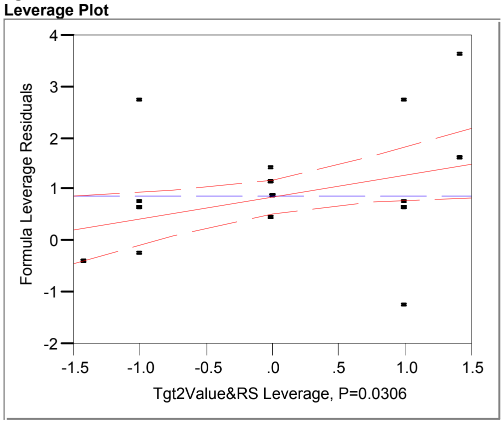
**Effect Tests**

| Source | Nparm | DF | Sum of Squares | F Ratio | Prob > F |
|---|---|---|---|---|---|
| Tgt2Value&RS | 1 | 1 | 4.432191 | 5.2102 | 0.0306 |
| SearchWeight&RS | 1 | 1 | 21.849453 | 25.6847 | <.0001 |
| Tgt2Value*Tgt2Value | 1 | 1 | 6.250000 | 7.3471 | 0.0115 |
| SearchWeight*SearchWeight | 1 | 1 | 4.694444 | 5.5185 | 0.0264 |

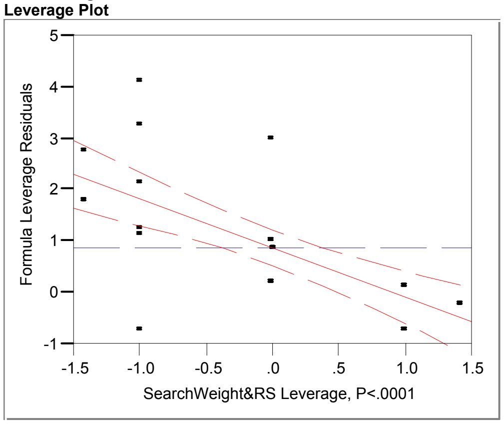
**Residual by Predicted Plot**



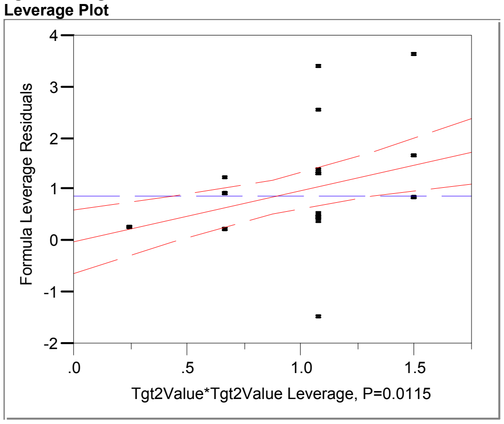

**Tgt2Value&RS**
**Leverage Plot**



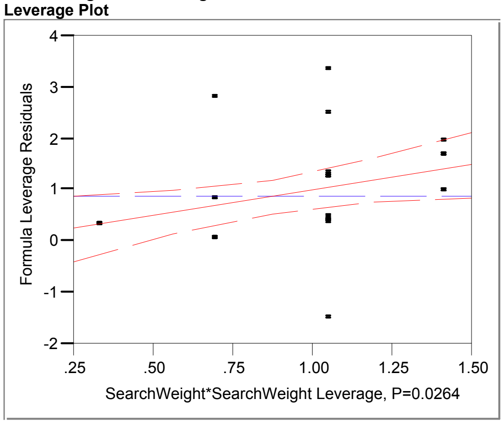

**SearchWeight&RS**
**Leverage Plot**



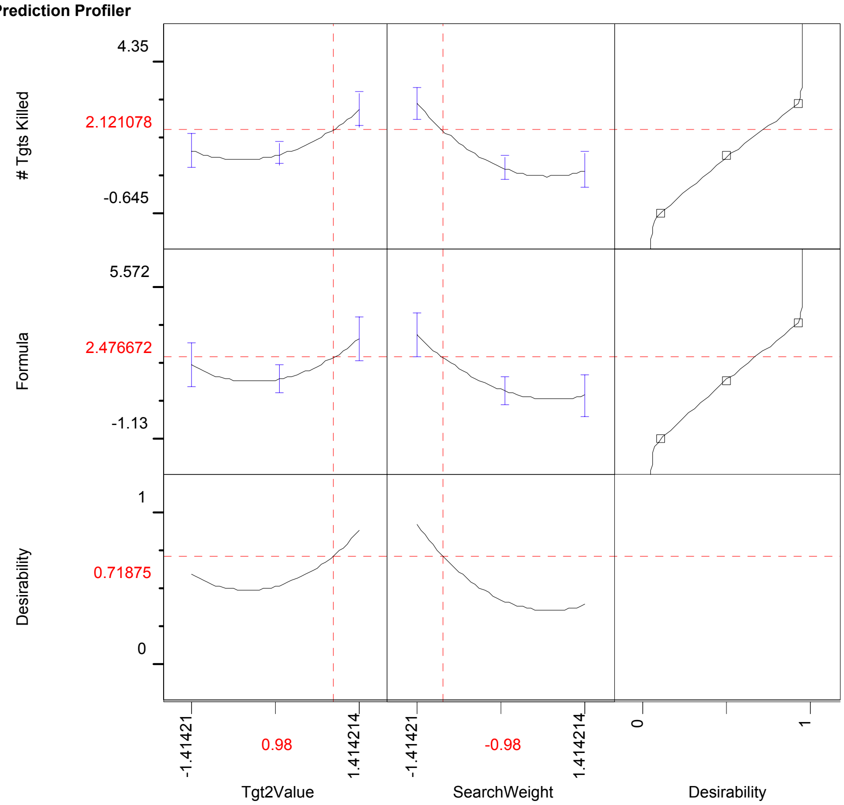
**Tgt2Value*Tgt2Value**
**Leverage Plot**




**SearchWeight*SearchWeight**
**Leverage Plot**




**Prediction Profiler**



# Bibliography


1. Dorigo, M. "Inspiration for Optimization from Social Insect Behaviour," Nature, vol 406: 39-42 (6 July 2000).

2. Frelinger, David, Joel Kvitky, and William Stanley. "Proliferated Autonomous Weapons; An Example of Cooperative Behavior." Technical Report, RAND, 1998.

3. Gillen, Daniel P. "Cooperative Behavior Schemes for Improving the Effectiveness of Autonomous Wide Area Search Munitions," AFIT Masters Thesis, 2001.

4. Jacques, David R. "Distributed Cooperation and Control for Autonomous Air Vehicles," Presentation to the Dayton Area Graduate Studies Institute, August 2001.

5. Jacques, David R. "Search Allocation, Classification, and Attack Decision Algorithms for Cooperative Wide Area Search Munitions," Workshop on Cooperative Control and Optimization, Gainsville, Florida, November 2001.

6. Jacques, David R. "Wide Area Search Munition Effectiveness and the Implications for Cooperative Behavior: Rough Draft." Work in Progress.

7. Jacques, David R. and Robert LeBlanc. "Effectiveness Analysis for Wide Area Search Munitions," American Institute of Aeronautics and Astronautics, Missile Sciences Conference (1998). Monterey, CA.

8. Kube, Ronald C. and Hong Zhang. "Collective Robots: From Social Insects to Robots," Adaptive Behavior, 2(2):189-218 (1993).

9. Myers, Raymond H. and Douglas C. Montgomery. Response Surface Methodology: Process and Product Optimization Using Designed Experiments. Wiley Series in Probability and Statistics, New York: Wiley-Interscience, 1995.

10. Neter et al. Applied Linear Statistical Models, 4$^{th}$ ed. Boston: McGraw-Hill, 1996.

11. Reynolds, C.W. "Flocks, Herds, and Schools: A Distributed Behavioral Model," Computer Graphics, 21(4):25-34 (July 1987).

12. Schumacher, Corey, Phillip R. Chandler, and Steven R. Rasmussen. "Task Allocation for Wide Area Search Munitions via Network Flow Optimization," American Institute of Aeronautics and Astronautics, 2001-4147 (2001).

13. Stone, Peter, and Manuela Veloso. "Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork," Artificial Intelligence, 1999.

14. Washburn, Alan R. Search and Detection, 2nd edition. Operations Research Society of America, 1989.

| REPORT DOCUMENTATION PAGE | *Form Approved*<br>*OMB No. 074-0188* |
|---|---|



| **1. REPORT DATE** *(DD-MM-YYYY)* | **2. REPORT TYPE** | **3. DATES COVERED** *(From – To)* |
|---|---|---|
| 26-03-2002 | **Master's Thesis** | Sep 2000 – Mar 2002 |

**4. TITLE AND SUBTITLE**

INVESTIGATION OF COOPERATIVE BEHAVIOR IN AUTONOMOUS WIDE AREA SEARCH MUNITIONS

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**

Dunkel, Robert E. III, Captain, USAF

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**

**7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)**

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 P Street, Building 640
WPAFB OH 45433-7765

**8. PERFORMING ORGANIZATION REPORT NUMBER**

AFIT/GAE/ENY/02-4

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

AFRL/MNGN
Attn: Mr. Rob Murphey
101 W Eglin Blvd. Ste 330
Eglin AFB, FL 32542-6810
(850) 882-2961 x3453

**10. SPONSOR/MONITOR'S ACRONYM(S)**

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

**12. DISTRIBUTION/AVAILABILITY STATEMENT**

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**

The purpose of this research is to investigate the effectiveness of wide-area search munitions in various scenarios using different cooperative behavior algorithms. The general scenario involves multiple autonomous munitions searching for an unknown number of targets of different priority in unknown locations. Three cooperative behavior algorithms are used in each scenario: no cooperation, cooperative attack only, and cooperative classification and attack. In the cooperative cases, the munitions allocate tasks on-line as a group, using linear programming techniques to determine the optimum allocation. Each munition provides inputs to the task allocation routine in the form of probabilities of successfully being able to complete the various tasks. These probabilities of success are based on statistical Poisson field theory. Weighting parameters are applied to the probabilities of success so that optimum settings can be determined via Response Surface Methodology.

Results are compared within and across the various scenarios. Initial results did not reflect expected behavior (due to poor choice of responses to optimize). Experiments were modified and more desirable results obtained. In general, cooperative engagement alone attacks and kills fewer targets than no cooperation. Cooperative classification however, kills fewer targets at low false target attack rates (< 0.005/sq km), but outperforms the other algorithms as the false target attack rate increases. This is due primarily to the fact that cooperative classification significantly reduces and stabilizes the effective false target attack rate.

**15. SUBJECT TERMS**

**Cooperative Engagement, Cooperative Behavior, Autonomous Munitions, Wide Area Search Munitions**

| **16. SECURITY CLASSIFICATION OF:** | | | **17. LIMITATION OF ABSTRACT** | **18. NUMBER OF PAGES** | **19a. NAME OF RESPONSIBLE PERSON** |
|---|---|---|---|---|---|
| **a. REPORT** | **b. ABSTRACT** | **c. THIS PAGE** | | | Lt Col David R. Jacques, AFIT/ENY |
| U | U | U | UU | 103 | **19b. TELEPHONE NUMBER** *(Include area code)*<br>(937) 255-3636, ext 4723; e-mail: David.Jacques@afit.edu |

**Standard Form 298 (Rev. 8-98)**
Prescribed by ANSI Std. Z39-18